SIEMENS

TRANSDATA
Data Display Terminals
Functional Description

User's Guide

Edition July 1990

Order-No. U6215-J-Z75-1-7600
Printed in the Federal Republic of Germany
1190AG 990.4 (1490)

A ring binder for this manual
is available at a cost of DM 5,–.
Order number: U3712-J-Z18-1

TRANSDATA
Data Display Terminals
Functional Description
User's Guide

Edition July 1990

Order-No. U6215-J-Z75-1-7600
Printed in the Federal Republic of Germany
1190 AG 990.4 (1490)

The reproduction, transmission or use of this document or its contents is not permitted without express written authority.

Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

Delivery subject to availability;
right of technical modifications reserved.

© Siemens AG 1990

Published by Bereich
Daten- und Informationstechnik
Postfach 83 09 51, D-8000 München 83

Siemens Aktiengesellschaft

Preface

This manual describes the functional characteristics and connection capabilities of data display terminals, and also operation with connected printers.

A second manual, entitled "TRANSDATA Data Display Terminals, Code Tables" [1], describes all the codes required for programming data display terminals and for data exchange with printers.

For whom is this manual designed?

The contents of this manual are intended for the programmer wishing to

- program outputs to data display terminals and printers,
- correctly interpret and evaluate inputs from data display terminals and printers.

The manual provides an immediate overview for interested persons of the options available for physical programming of data display terminal functions.

What do you need to know?

Previous experience of programming data display terminals and printers is necessary.

Note

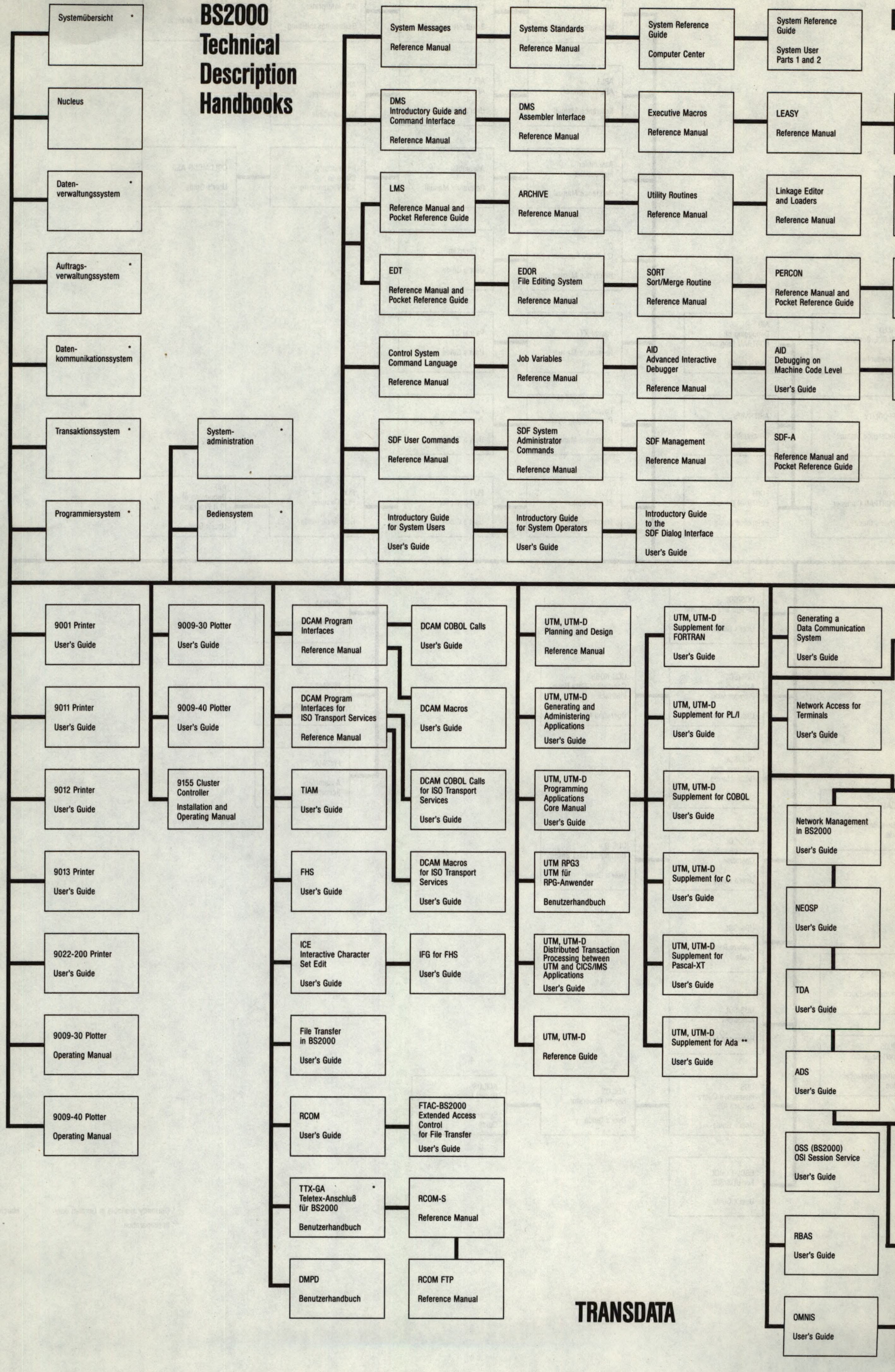
Programming of data display terminal functions in application programs can be carried out only in the manner described in this manual.

An appeal to the reader

Please let us know what you think of this manual. If you are satisfied, then we know we are on the right track; if not, let us know where the stumbling blocks are so that we can take steps to eliminate them and improve our documentation. Reply forms are provided at the end of the manual for this purpose.

User Documentation Department DI ST QM 2

BS2000 Technical Description Handbooks



- Systemübersicht *
- Nucleus
- Datenverwaltungssystem *
- Auftragsverwaltungssystem *
- Datenkommunikationssystem *
- Transaktionssystem *
- Programmiersystem *
- Systemadministration *
- Bediensystem *

- System Messages Reference Manual
  - Systems Standards Reference Manual
    - System Reference Guide Computer Center
      - System Reference Guide System User Parts 1 and 2
- DMS Introductory Guide and Command Interface Reference Manual
  - DMS Assembler Interface Reference Manual
    - Executive Macros Reference Manual
      - LEASY Reference Manual
- LMS Reference Manual and Pocket Reference Guide
  - ARCHIVE Reference Manual
    - Utility Routines Reference Manual
      - Linkage Editor and Loaders Reference Manual
- EDT Reference Manual and Pocket Reference Guide
  - EDOR File Editing System Reference Manual
    - SORT Sort/Merge Routine Reference Manual
      - PERCON Reference Manual and Pocket Reference Guide
- Control System Command Language Reference Manual
  - Job Variables Reference Manual
    - AID Advanced Interactive Debugger Reference Manual
      - AID Debugging on Machine Code Level User's Guide
- SDF User Commands Reference Manual
  - SDF System Administrator Commands Reference Manual
    - SDF Management Reference Manual
      - SDF-A Reference Manual and Pocket Reference Guide
- Introductory Guide for System Users User's Guide
  - Introductory Guide for System Operators User's Guide
    - Introductory Guide to the SDF Dialog Interface User's Guide

- 9001 Printer User's Guide
- 9011 Printer User's Guide
- 9012 Printer User's Guide
- 9013 Printer User's Guide
- 9022-200 Printer User's Guide
- 9009-30 Plotter Operating Manual
- 9009-40 Plotter Operating Manual

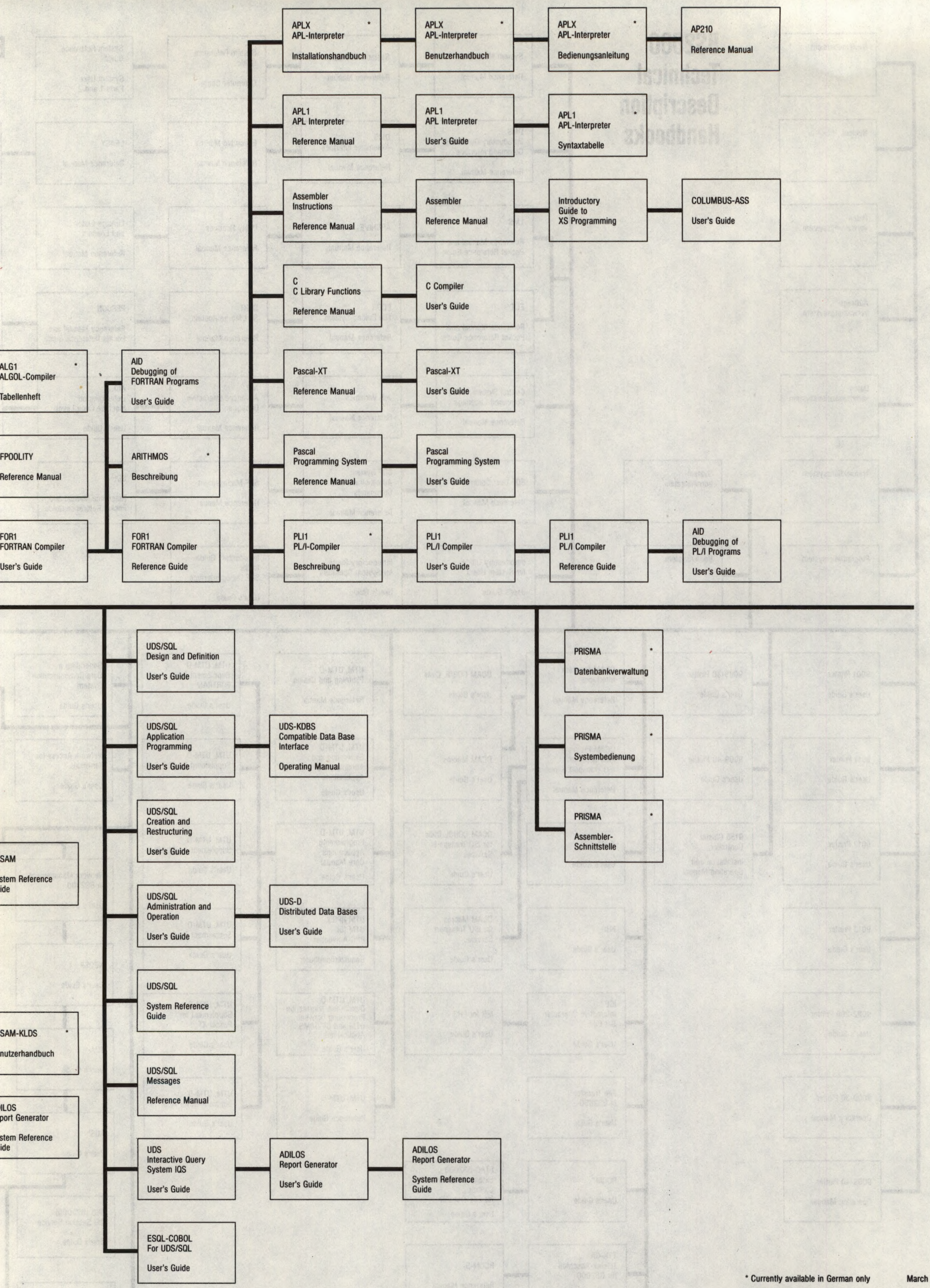
- 9009-30 Plotter User's Guide
- 9009-40 Plotter User's Guide
- 9155 Cluster Controller Installation and Operating Manual

- DCAM Program Interfaces Reference Manual
  - DCAM COBOL Calls User's Guide
  - DCAM Macros User's Guide
- DCAM Program Interfaces for ISO Transport Services Reference Manual
  - DCAM COBOL Calls for ISO Transport Services User's Guide
  - DCAM Macros for ISO Transport Services User's Guide
- TIAM User's Guide
- FHS User's Guide
- ICE Interactive Character Set Edit User's Guide
  - IFG for FHS User's Guide
- File Transfer in BS2000 User's Guide
- RCOM User's Guide
  - FTAC-BS2000 Extended Access Control for File Transfer User's Guide
- TTX-GA Teletex-Anschluß für BS2000 * Benutzerhandbuch
  - RCOM-S Reference Manual
    - RCOM FTP Reference Manual
- DMPD Benutzerhandbuch

- UTM, UTM-D Planning and Design Reference Manual
- UTM, UTM-D Generating and Administering Applications User's Guide
- UTM, UTM-D Programming Applications Core Manual User's Guide
  - UTM, UTM-D Supplement for FORTRAN User's Guide
  - UTM, UTM-D Supplement for PL/I User's Guide
  - UTM, UTM-D Supplement for COBOL User's Guide
  - UTM, UTM-D Supplement for C User's Guide
  - UTM, UTM-D Supplement for Pascal-XT User's Guide
  - UTM, UTM-D Supplement for Ada ** User's Guide
- UTM RPG3 UTM für RPG-Anwender Benutzerhandbuch
- UTM, UTM-D Distributed Transaction Processing between UTM and CICS/IMS Applications User's Guide
- UTM, UTM-D Reference Guide

- Generating a Data Communication System User's Guide
- Network Access for Terminals User's Guide
- Network Management in BS2000 User's Guide
- NEOSP User's Guide
- TDA User's Guide
- ADS User's Guide
- OSS (BS2000) OSI Session Service User's Guide
- RBAS User's Guide
- OMNIS User's Guide

TRANSDATA



APLX APL-Interpreter * Installationshandbuch

APLX APL-Interpreter * Benutzerhandbuch

APLX APL-Interpreter * Bedienungsanleitung

AP210 Reference Manual

APL1 APL Interpreter Reference Manual

APL1 APL Interpreter User's Guide

APL1 APL-Interpreter Syntaxtabelle

Assembler Instructions Reference Manual

Assembler Reference Manual

Introductory Guide to XS Programming

COLUMBUS-ASS User's Guide

C C Library Functions Reference Manual

C Compiler User's Guide

ALG1 * ALGOL-Compiler Tabellenheft

AID Debugging of FORTRAN Programs User's Guide

Pascal-XT Reference Manual

Pascal-XT User's Guide

FPOOLITY Reference Manual

ARITHMOS * Beschreibung

Pascal Programming System Reference Manual

Pascal Programming System User's Guide

FOR1 FORTRAN Compiler User's Guide

FOR1 FORTRAN Compiler Reference Guide

PLI1 * PL/I-Compiler Beschreibung

PLI1 PL/I Compiler User's Guide

PLI1 PL/I Compiler Reference Guide

AID Debugging of PL/I Programs User's Guide

UDS/SQL Design and Definition User's Guide

PRISMA * Datenbankverwaltung

UDS/SQL Application Programming User's Guide

UDS-KDBS Compatible Data Base Interface Operating Manual

PRISMA * Systembedienung

SAM stem Reference ide

UDS/SQL Creation and Restructuring User's Guide

PRISMA * Assembler-Schnittstelle

UDS/SQL Administration and Operation User's Guide

UDS-D Distributed Data Bases User's Guide

UDS/SQL System Reference Guide

SAM-KLDS * nutzerhandbuch

UDS/SQL Messages Reference Manual

ILOS port Generator stem Reference ide

UDS Interactive Query System IQS User's Guide

ADILOS Report Generator User's Guide

ADILOS Report Generator System Reference Guide

ESQL-COBOL For UDS/SQL User's Guide

* Currently available in German only

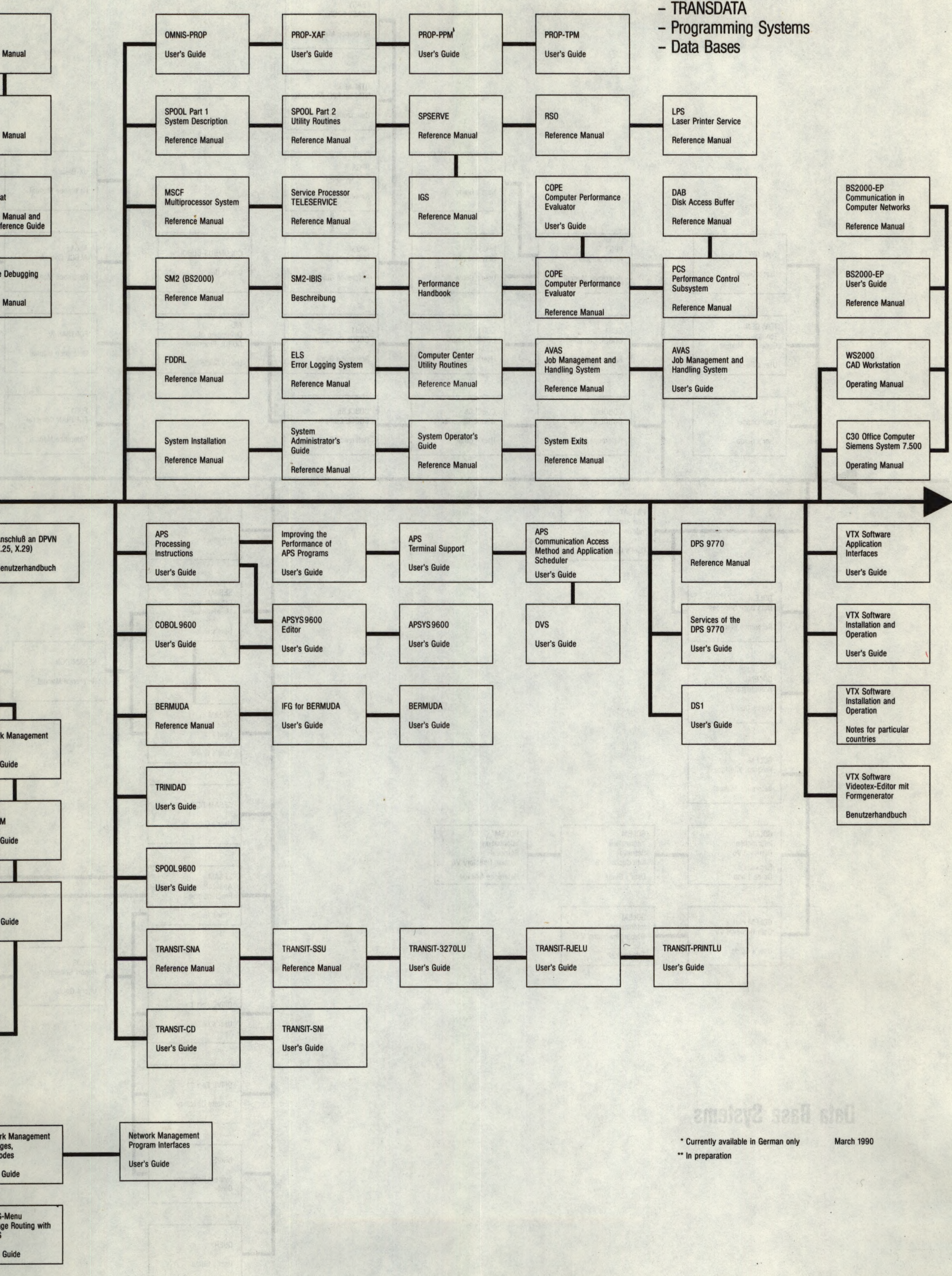
** In preparation

March 1990

00

Publications Plan for

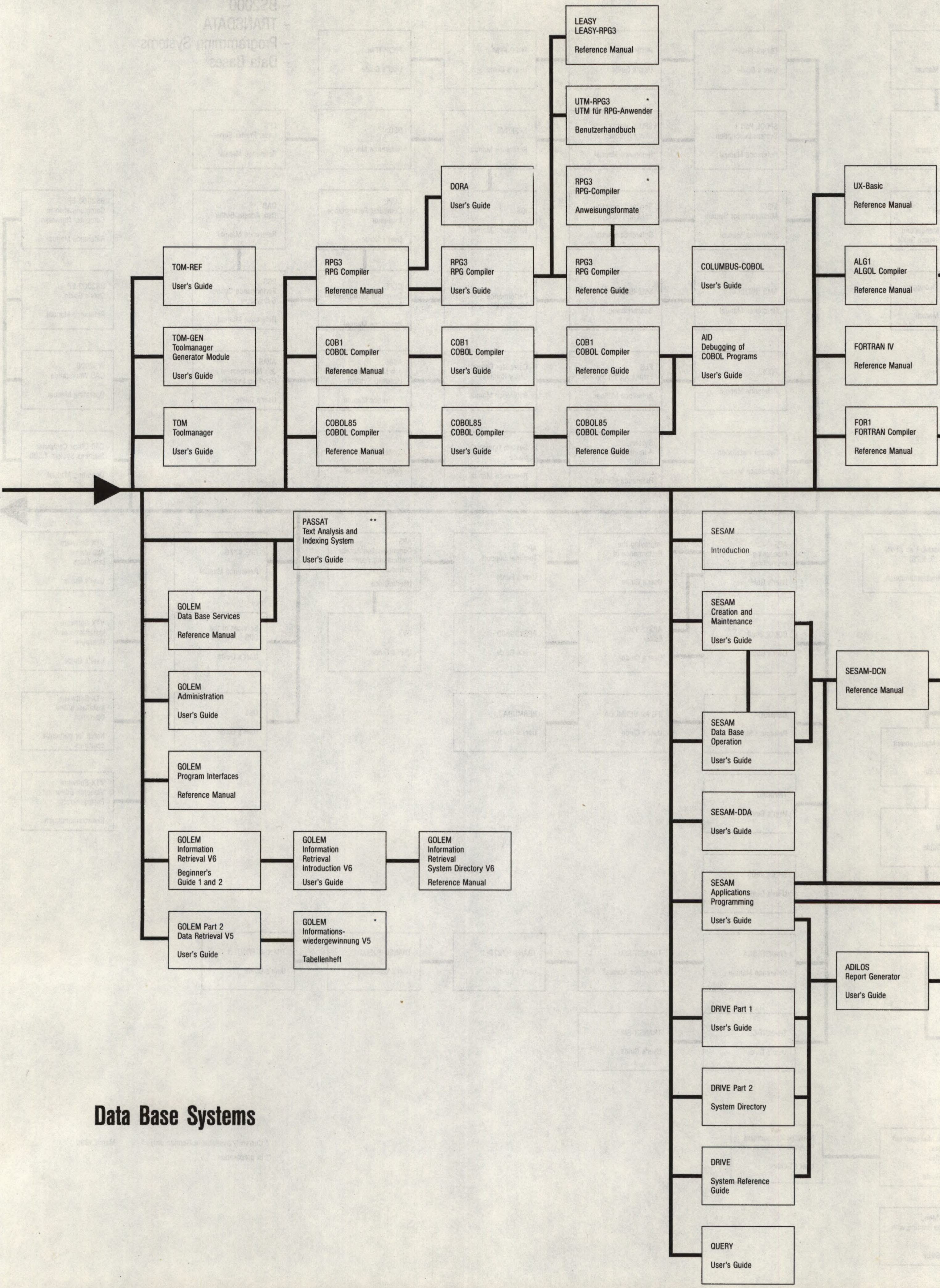
- BS2000
- TRANSDATA
- Programming Systems
- Data Bases



* Currently available in German only
** In preparation

March 1990

Programming Systems



TOM-REF
User's Guide

TOM-GEN
Toolmanager
Generator Module
User's Guide

TOM
Toolmanager
User's Guide

RPG3
RPG Compiler
Reference Manual

COB1
COBOL Compiler
Reference Manual

COBOL85
COBOL Compiler
Reference Manual

DORA
User's Guide

RPG3
RPG Compiler
User's Guide

COB1
COBOL Compiler
User's Guide

COBOL85
COBOL Compiler
User's Guide

LEASY
LEASY-RPG3
Reference Manual

UTM-RPG3 *
UTM für RPG-Anwender
Benutzerhandbuch

RPG3 *
RPG-Compiler
Anweisungsformate

RPG3
RPG Compiler
Reference Guide

COB1
COBOL Compiler
Reference Guide

COBOL85
COBOL Compiler
Reference Guide

COLUMBUS-COBOL
User's Guide

AID
Debugging of
COBOL Programs
User's Guide

UX-Basic
Reference Manual

ALG1
ALGOL Compiler
Reference Manual

FORTRAN IV
Reference Manual

FOR1
FORTRAN Compiler
Reference Manual

Data Base Systems

PASSAT **
Text Analysis and
Indexing System
User's Guide

GOLEM
Data Base Services
Reference Manual

GOLEM
Administration
User's Guide

GOLEM
Program Interfaces
Reference Manual

GOLEM
Information
Retrieval V6
Beginner's
Guide 1 and 2

GOLEM
Information
Retrieval
Introduction V6
User's Guide

GOLEM
Information
Retrieval
System Directory V6
Reference Manual

GOLEM Part 2
Data Retrieval V5
User's Guide

GOLEM *
Informations-
wiedergewinnung V5
Tabellenheft

SESAM
Introduction

SESAM
Creation and
Maintenance
User's Guide

SESAM
Data Base
Operation
User's Guide

SESAM-DDA
User's Guide

SESAM
Applications
Programming
User's Guide

SESAM-DCN
Reference Manual

ADILOS
Report Generator
User's Guide

DRIVE Part 1
User's Guide

DRIVE Part 2
System Directory

DRIVE
System Reference
Guide

QUERY
User's Guide

Contents

1 Introduction 1-1
1.1 TRANSDATA Concept 1-1
1.1.1 TRANSDATA Hardware 1-1
1.1.2 TRANSDATA Software 1-2
1.2 Data Display Terminal Concept 1-3
1.2.1 Basic Requirements 1-3
1.2.2 Compatibility 1-3
1.2.3 Operating Modes 1-3
Unformatted Data Display Terminal Operation 1-3
Formatted Data Display Terminal Operation 1-4

2 Connection Options for Data Display Terminals 2-1
2.1 Standalone Terminals 2-1
2.2 Cluster Terminals 2-1

3 Message Traffic involving Data Display Terminals 3-1
3.1 Data Display Terminal for Standalone Connection (DDT-S) 3-1
3.1.1 MSV1 (Synchronous Procedure) 3-1
3.1.2 HDLC Procedure 3-1
3.2 Data Display Terminal for Cluster Connection (DDT-C) 3-1
3.2.1 Message Traffic between CPU/Communication Computer and Cluster Controller 3-2
3.2.2 Message Traffic between Cluster/Local Line Controller and DDT-C 3-2
3.2.2.1 Data Output to a Data Display Terminal 3-3
3.2.2.2 Data Input from a Data Display Terminal 3-3
3.2.3 Message Traffic between Communication Computer and DDT-C using HDLC 3-3

4 Message Structure for Data Display Terminals 4-1
4.1 Message Prefix 4-1
4.1.1 Operation with Message Header for Data Output to a Data Display Terminal 4-1
4.1.2 Operation with Message Header for Data Input from a Data Display Terminal 4-2
Message Header for Data Input from a Data Display Terminal 4-2
Functions of Parameter Areas 4-2
Structure of Parameter Area PAR 00E 4-3
Structure of Parameter Area PAR 01E 4-3
Text Identifier 4-4
Error Messages 4-4
4.1.3 Message Prefix for Operation with Acknowledgments 4-5
4.2 Message Section 4-6
4.2.1 Message Section with Parameter Entries (PAG) 4-6
Message Structure with Parameter Entries (PAG) 4-7
Structure of Parameter Entries 4-7
Functions of Parameter Areas and Parameter Segments 4-8
Possible Errors 4-9
4.2.1.1 Structure of Parameter Area PAR 00L 4-9
Send Data in Formatted Mode 4-10
Handling the MOD and BLI Bits 4-10
4.2.1.2 Structure of Parameter Area PAR 01L 4-10
4.2.1.3 Structure of Parameter Area PAR 00D 4-11
4.2.2 Message Section with Structured Commands 4-12
4.2.2.1 Read SIDATA Parameters (LSID) 4-13
4.2.2.2 Read P-Areas (LPBE) 4-13

4.2.2.3 Load Character Generator (LZEG) . . . 4-14
Editing of Data for Loading Fonts . . . 4-16
Loading Character Boxes without Compression . . . 4-17
Loading Character Boxes with Compression . . . 4-18
Producing the Compressed Bit String . . . 4-18
Coding of Data in Data Field for Data Transmission . . . 4-24
4.2.2.4 Define Screen (DEBS) . . . 4-25
4.2.2.5 Reset Screen (RDSS) . . . 4-27
4.2.2.6 Write Display Area (SRBR) . . . 4-27
Data Transmission to DDT . . . 4-28
Data Transmission to CPU . . . 4-28
Coding for Transmission with Expanded Mode . . . 4-29
4.2.2.7 Error Message for Structured Commands . . . 4-29
4.2.3 Message Section with Positioning Commands (IS4 Sequence) . . . 4-30
Structure of Positioning Commands . . . 4-30
4.2.4 Message Section with Field Separators and Attributes (IS2, IS3 Sequences) . . . 4-31
Structure of Character String with Field Separators . . . 4-32
4.2.4.1 Entering Field Separators . . . 4-32
4.2.4.2 Modifying Field Separators . . . 4-32
4.2.4.3 Erasing Field Separators . . . 4-32
4.2.5 Message Section with Field and Character Attributes . . . 4-33
4.2.5.1 Field Attributes (FAT) . . . 4-34
Field Attribute 1 . . . 4-34
Field Attribute 2 . . . 4-36
Field Attribute 3 . . . 4-36
Field Attribute 4 . . . 4-36
Transmission of Field Attributes to CPU . . . 4-38
4.2.5.2 Create Field Attributes (EZFA) . . . 4-38
4.2.5.3 Modify Field Attributes (MOFA) . . . 4-39
4.2.5.4 Character Attributes . . . 4-40
Transmission of Character Attributes to DDT . . . 4-41
Transmission of Character Attributes to CPU . . . 4-41
Reset Character Attribute (RSZA) . . . 4-41
Define Display Attribute (DEAA) . . . 4-42
Define Color Attributes (DEFA) . . . 4-42
Define Font (DEZS) . . . 4-43
4.2.6 Message Section with Device Functions (ESC Sequences) . . . 4-44
Structure of Character String for Device Functions in Text . . . 4-45
4.2.7 Description of Device Functions . . . 4-46
Cursor Positioning Functions . . . 4-46
Text Manipulation Functions . . . 4-48
Erase Functions . . . 4-51
Data Transmission Functions . . . 4-53
Functions for Data Exchange between DDT and Centrally Connected Printer . . . 4-54
Programmable Key Functions . . . 4-55
Special Functions . . . 4-56
4.3 Message Structure for Output of a System Message . . . 4-58
4.4 Operation with Status Messages . . . 4-60
4.4.1 Query . . . 4-60
4.4.2 Query Reply . . . 4-60
4.5 Message Structure for Operation with ID Card Reader . . . 4-61
4.5.1 System Request for ID Card Information . . . 4-61
4.5.2 ID Card Format . . . 4-61
4.5.3 Reading ID Card Information . . . 4-61
Reading Errored ID Card Information . . . 4-61
4.5.4 Error Messages . . . 4-62

4.6 Chip Card Terminal (CKT) Connection 4-63
4.6.1 Data Transmission from CPU to CKT 4-64
4.6.2 Data Transmission from CKT to CPU 4-65

5 Printer Operation 5-1
5.1 Configuration 5-1
5.2 Operating Modes 5-2
Unsolicited Print Output 5-2
Hardcopy Mode 5-3
5.3 Acknowledgments 5-4
Table of Functions for Data Display Terminals 5-5

Abbreviations
References
Index

1 Introduction

The course of technological development has supported the trend toward decentralization of processing power. This decentralization has been implemented by Siemens in the form of the hardware and software components of the Siemens TRANSDATA System.

1.1 TRANSDATA Concept

1.1.1 TRANSDATA Hardware

A large range of host computers and an extensive spectrum of compatible TRANSDATA hardware products [2] allow the implementation of many types of network.

The following TRANSDATA hardware products are available:

- front-end processors (FEP), compact front-end processors (CFEP),
- remote front-end processors (FEPR),
- terminal computers (TC) with local peripherals and products of the TRANSDATA 9.600 System,
- terminals,
- communications features
  - cluster controllers (LCC, RCC, CC),
  - interface expanders (IE),
  - concentrator (MSC),
  - protocol converter concentrator (PCC).

1.1.2 TRANSDATA Software

The TRANSDATA software consists of two compatible components

- PDN (program system for teleprocessing and network control) and
- DCM (Data Communication Methods),

specially designed to complement one another. They handle the transport of data between sender and receiver in the data communication system, control and check the actual transfer of data, and edit the data in a form convenient to the user.

Communication computers of the TRANSDATA 960 System use the PDN operating system.

Host computers are driven by the Data Communication Methods DCM.

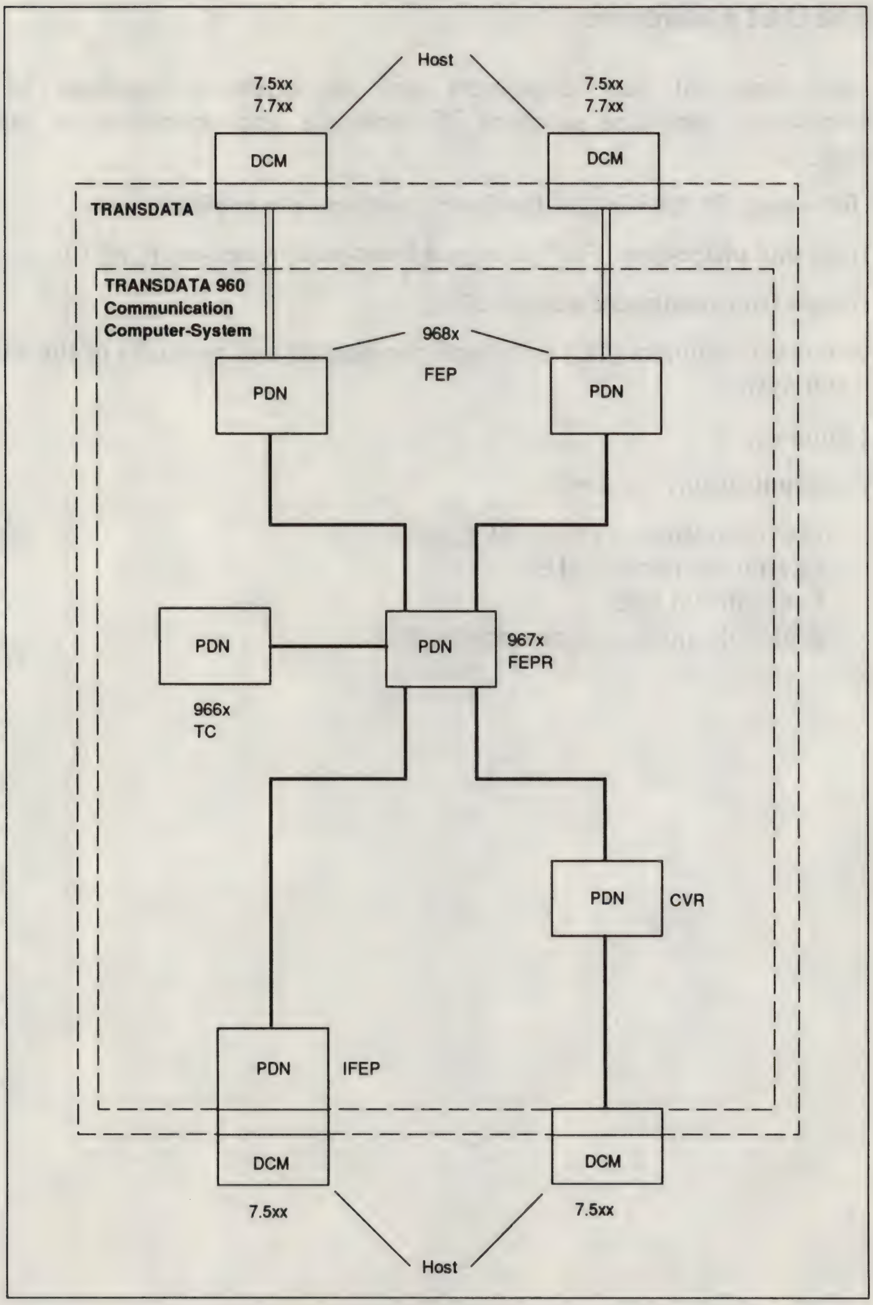


Fig. 1-1 TRANSDATA software

1.2 Data Display Terminal Concept

1.2.1 Basic Requirements

Data display terminals are assessed on the basis of performance, cost effectiveness and ergometric considerations. The SIEMENS data display terminals satisfy these basic requirements:

- Functions and peripherals are optimally suited to interactive and data entry tasks.
- Ergonometric requirements and standards have been fully implemented in equipment design.
- Freedom to configure many types of data network is provided.
- Integration in existing applications and adaptability to any given organizational conditions are assured.

1.2.2 Compatibility

The variants of the 9746,47,48 Data Display Terminals are compatible with the functions of the 9750/49 Data Display Terminals in all hardware and software configurations.

The 9752 color terminal is compatible with the 9749 and 9750 Data Display Terminals as far the system is concerned. The central software can interoperate simultaneously with the 9746, 9747, 9748 and 9752 Data Display Terminals without modification.

The 9755,9756,9758M,9762C,9763M/C terminals are provided with a modified and expanded function set. In their basic functions they correspond to the 9750/9749. If the extended capabilities are used, these terminals must be generated as a separate device type.

All the data display terminals referred to here use the 810 protocol.

Note

> The appendix includes a table showing functions available on individual data display terminals.

1.2.3 Operating Modes

Interactive operation with data display terminals can take place in either

- unformatted or
- formatted mode.

Unformatted Data Display Terminal Operation

When a data display terminal user is working with the communication application `$DIALOG`, data exchange with the operating system (commands, messages) takes place in unformatted mode. Utilities and application programs can also operate in unformatted mode with data display terminals.

Formatted Data Display Terminal Operation

Application programs can work in formatted mode with data display terminals. In formatted mode, the screen of the terminal is divided up into fields. The following field characteristics can be assigned to these fields, for example:

- protected / unprotected
- printable
- markable
- modified
- numeric / alphanumeric
- protected transmittable / protected non-transmittable

2 Connection Options for Data Display Terminals

The data display terminals may be classified as

- standalone terminals (DDT-S) and
- cluster terminals (DDT-C).

2.1 Standalone Terminals

Standalone terminals can be connected to computers either directly or via public/private networks.

- Public leased-circuit data network
- Switched telephone network
- Datex network (Datex-L or Datex-P, with or without PCC for synchronous data terminal equipments)
- Interface expander (IE)
- Concentrator (MSC)

2.2 Cluster Terminals

The DDT-C is suitable for connection to inhouse networks utilizing WTÜ (coded di-phase) transmission or to alternating pulse edge 2-wire transmission networks and is connected to

- cluster controllers, using WTÜ (coded di-phase), such as
  - 8170 LCC (local connection)
  - 8171 RCC (remote connection)
  - 9155 CC (remote and/or inhouse connection)
- communication computers of the TRANSDATA 960 System with local line controller (BAST, controller for BAM lines),
- host computers (including IFEPs) via the integrated terminal controller (ITC).
- host computers (including IFEPs) via double block buffer with alternating pulse edge 2-wire adapter or primary.

3 Message Traffic involving Data Display Terminals

Agreed transmission rules (procedures) are prerequisite for message traffic.

3.1 Data Display Terminal for Standalone Connection (DDT-S)

With data display terminals for standalone connection (standalone terminals), messages are transmitted using the MSV or HDLC procedure:

3.1.1 MSV1 (Synchronous Procedure)

Before a message can be passed to a system component, the device to be addressed must be activated by the specification of control characters and addresses. Here the CPU has the status of a control station. Only the CPU initiates data transfer operations; it does this by sending

- selecting signals (EA) for data output or
- polling signals (SA) for data input

(differentiated by bit 1 of AD1/AD2).

3.1.2 HDLC Procedure

The HDLC procedure is a code-transparent, standardized synchronous transmission procedure with distinctive security elements.

3.2 Data Display Terminal for Cluster Connection (DDT-C)

With data display terminals for cluster connection, the message traffic is handled using the BAM procedure:

- between data display terminals and cluster controllers (8170 LCC, 8171 RCC, 9155 CC)
- between data display terminals and local line controllers

using the HDLC procedure:

- between data display terminals and alternating pulse edge adapter (in a communication computer)
- between data display terminals and primary.

3.2.1 Message Traffic between CPU/Communication Computer and Cluster Controller

- Message traffic between CPU and 8170 Local Cluster Controller.

  The cluster controller is connected directly by means of a standard interface cable up to 30 meters in length to a trunk on the byte multiplexer channel of a CPU.

  The message traffic is handled in accordance with the procedure for the CPU standard interface, including the following functions:

  - signal timing for information transfer on the data lines in byte multiplex operation,
  - interpretation of the CPU commands in the cluster controller,
  - generation and interpretation of status indicators,
  - monitoring of vertical parity in the CPU.

- Message traffic between CPU/communication computer and 8171 Remote Cluster Controller.

  The message traffic over one (or two) long-distance line(s) is handled in accordance with the MSV1 synchronous procedure with error recovery.

- Message traffic between CPU/communication computer and 9155 Cluster Controller for remote or inhouse connection.

  The message traffic is handled via a long-distance or inhouse line using the HDLC synchronous procedure. The maximum data transfer rate is 64 kbps.

3.2.2 Message Traffic between Cluster/Local Line Controller and DDT-C

The message traffic between cluster controller or local line controller and data display terminal is handled in accordance with a special internal system procedure (BAM procedure), where the initiative lies with the cluster controller.

Under the control of commands from the cluster controller, the data display terminal will be in one of the three procedure states:

- initial state,
- input state,
- output state.

The reaction of a data display terminal depends on its current procedure state. On the basis of the acknowledgments sent by a data display terminal, the cluster controller can recognize whether the data display terminal is in the correct procedure state and whether it has correctly executed commands that it has received.

3.2.2.1 Data Output to a Data Display Terminal

The cluster controller sends a write command to the data display terminal, causing the latter to switch to output state and respond with an acknowledgment. The cluster controller then forwards the data to be output by the CPU to the data display terminal. Each message character is acknowledged by the data display terminal.

3.2.2.2 Data Input from a Data Display Terminal

The cluster controller cyclically issues a command to each connected data display terminal. If a data display terminal has been placed in 'Send' status (e.g. by means of the 'DÜ1' key), then it will respond to the next interrogation command with a modified status byte.
The cluster controller then sends a read command to this data display terminal; the data display terminal switches to input state and acknowledges with a status byte.
The cluster controller then issues further commands to request the message characters individually from the data display terminal; the response to each command is one message character. The cluster controller forwards the message characters it receives directly to the CPU.

3.2.3 Message Traffic between Communication Computer and DDT-C using HDLC

The message traffic between a primary or a multiplexer and a DDT-C is handled by means of the standardized transparent HDLC unbalanced transmission procedure, where the coding employed complies with EBCDIC DF03.

4 Message Structure for Data Display Terminals

A message comprises the following components:

- message prefix and
- message section.

The table below provides an overview of the various options available for the message structure.

Meaning of abbreviations used in the table:

NBZ = start-of-message character
STA = structured commands
NT = message text
PAG = parameter entries

Message prefix		Message section	Remarks
NBZ	–	NT	e.g. simple dialog
NBZ	m.hdr	NT	e.g. loading parameters
NBZ	–	nPAG + STA	
NBZ	–	nPAG + NT + STA	e.g. message with special commands
NBZ	–	NT + nPAG + STA	
NBZ	–	nPAG + NT + nPAG + STA	
NBZ	–	NT + nPAG + NT + STA	

Table 4-1 Message structure

4.1 Message Prefix

The message prefix is present for every input and output. It comprises at least the start-of-message character (NBZ).

4.1.1 Operation with Message Header for Data Output to a Data Display Terminal

Note

> The use of message headers during data output is not permitted for new applications. Control of the functions is effected by means of parameter entries (PAG) (see section 4.2.1). These are not part of the message prefix but of the actual message itself! For processing of messages with a message header, see [1].

4.1.2 Operation with Message Header for Data Input from a Data Display Terminal

In the case of data input from a data display terminal (DDT = = > CPU), the message prefix must comprise at least the start-of-message character andan 8-byte message header (=PAR 00E).
If a text identifier is included, it follows the message header. The message header may also be longer than 8 bytes without the text identifier.
With field mode, the message header is 24 bytes long
(=PAR 00E + PAR 01E + PAR 02E).

Message structure

NBZ	Meaning
NBZ	Message header comprises PAR 00E
NBZ	Message header comprises PAR 00E + PAR 01E
NBZ	Message header comprises PAR 00E + PAR 01E + PAR 02E
NBZ	Message header comprises PAR 00E + PAR 01E + PAR 02E + PAR 03E
NBZ	Message header comprises PAR 00E + PAR 01E + PAR 03E

Table 4-2 Message header for data input

Message Header for Data Input from a Data Display Terminal

The message header forms part of the message prefix and may be up to 32 bytes long (24 bytes for field mode). The following parameter areas are transferred in this message header:

- PAR 00E,
- PAR 01E,
- PAR 02E,
- PAR 03E.

With each data transfer operation to the CPU, these parameter areas - depending on entries in PAR00L or PAR 01L - are prefixed to the message by the data display terminal.

Functions of Parameter Areas

PAR 00E:	This parameter area contains device-specific entries.
PAR 01E:	This parameter area is the extension of PAR 00E and is used for entry of the text header length, for specifying the cursor position prior to initiation of data transfer, and as an indicator for error messages.
PAR 02E:	This parameter area is reserved for field mode.
PAR 03E:	This parameter area is used for terminal identification. The terminal identification is taken from parameter area PAR 13D.

Structure of Parameter Area PAR 00E

Char.	Short name	Meaning
1	SAW	Depending on the DÜ key used, the corresponding send command SAW1 or SAW2 is entered in PAR 00E and executed.
2	PFLNG	Not permitted for new application
3	ZLNG	Not permitted for new application
4	AUSG	Revision level
5	ZZ1	Status character 1: ZZ1 contains info. such as - ID card inserted in reader - keylock switches On - local devices connected
6	ZZ2	Status character 2: Keyboard and local devices are connected to ports 1 through 5.
7	CDS	Code for send key. The code for the activated send key is entered in this character.
8	WAR	Queue identifier The DDT returns unchanged the WAR defined by the CPU.

Table 4-3 Description of characters in PAR 00E

Structure of Parameter Area PAR 01E

Char.	Short name	Meaning
1	TIL	Specifies if and in what length a text identifier TI is to be transferred from the mailbox of the DDT to the CPU.
2	TKL	Text header length, reserved
3	ZLA	Line address
4	SPA 1	Column address 1
5	SPA 2	Column address 2
6	FEM	Error message: The DDT notes in the FEM any non-observance of conventions (e.g. invalid coding for positioning received, deletion of non-existent field separators etc.)
7	—	Reserved
8	ZZ5	Status character: The last 8 characters of the message header contain the terminal identifier TID. This is taken from PAR 13D.

Table 4-4 Description of characters in PAR 01E

Text Identifier

The text identifier (TI) can be defined by the programmer. It is therefore possible, for example, for data inputs from different data display terminals to be given different text identifiers and thus have specific assignments.

The text identifier follows the message header.

A text identifier is stored by a data output in parameter segments PAR 01D, PAR 02D and PAR 03D (8 bytes each).

The text identifier length (TIL) specifies whether and in what length a text identifier from the parameter segments PAR 01D – PAR 03D of the data display terminal is to be transferred to the CPU.

If a text identifier is stored in the parameter segments, then - depending on character 1 (TIL) of PAR 01L - it will be transferred immediately following the message header to the CPU with each data input. The length of the text identifier is specified in PAR01E.

Error Messages

If the code conventions for the device functions are not observed by the CPU during a data output, then the corresponding error message is noted by the data display terminal in character 6 (FEM) of PAR 01E.
The data display terminal passes this parameter area to the CPU on request with the next data input.

4.1.3 Message Prefix for Operation with Acknowledgments

For operation with acknowledgments, the message comprises only the message prefix with a length of 4 bytes.

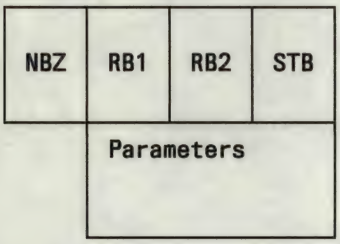


Fig. 4-1 Acknowledgments from locally connected devices

Start-of-message character (NBZ):	X'41': Positive acknowledgment X'42': Negative acknowledgment
Return bytes (RB1,RB2):	Taken from characters 6 and 7 (RB1,RB2) of PAR 10L/D through PAR 70L/D.
Status byte (STB):	Contains error codes.

4.2 Message Section

Message sections can consist of

- parameter entries (PAG) without message texts (loading of parameter areas and parameter segments),
- message texts without parameter entries,
- parameter entries and message texts (messages with special commands).

Message texts can be constructed using

- text, consisting of
  - numeric characters,
  - alphabetic characters (uppercase and lowercase),
  - special symbols,
- absolute positioning commands (IS4 sequences),
- field separators (IS2, IS3 sequences),
- structured commands,
- field and character attributes,
- device functions (ESC sequences).

4.2.1 Message Section with Parameter Entries (PAG)

The parameter entries (PAG) are not part of the message header. **The length of the PAGs is not given in the start-of-message character (NBZ); the NBZ is X'40' (message without message header).** The parameter entries are used to selectively load parameter areas and parameter segments.
Parameter areas and parameter segments are particular storage areas in the data display terminal that are used to control the operational characteristics of the data display terminal.

Parameter entries can be used for the entry of:

- device functions,
- send commands,
- text identifiers,
- operating states.

When parameters have been loaded, these parameters remain in effect until they are overwritten by new parameter entries or by a message header and until erased by activation of the device function 'LSP' (depending on character 3 of PAR 00L).
When the data display terminal is powered off, the stored parameters are erased; on powering on, certain basic information is loaded automatically into the data display terminal.

Message Structure with Parameter Entries (PAG)

Message prefix	Message section
NBZ	
NBZ	nPAG
NBZ	NT
NBZ	nPAG + NT
NBZ	NT + nPAG
NBZ	nPAG + NT + nPAG
NBZ	NT + nPAG + NT

Fig. 4-2 Structure of a message for data transfer from CPU to DDT

Meaning of abbreviations used in the table:

NBZ = start-of-message character
PAG = parameter entries (PARxyL or PARxyD where x = 0-7 and y = 0-3)
NT = message text

Structure of Parameter Entries

The parameter entries are announced by means of a parameter announcer (PAK). The following specifications for the addresses of

- parameter area and
- parameter segment

control the selective storage of parameter entries in the corresponding parameter areas and parameter segments of the data display terminal.

NBZ	ESC	Space	PAK	Address of parameter area	Address of parameter segment	Parameter entries 8 bytes	Parameter entries 8 bytes	Parameter entries 8 bytes	Parameter entries 8 bytes	Termination character

Fig. 4-3 Structure of parameter entries

Note

If different parameter areas are loaded in succession, then each parameter area must be announced by means of the parameter announcement (ESC, Space, PAK) and terminated by means of the termination character (!). The same also applies if text segments occur between parameter entries.

Functions of Parameter Areas and Parameter Segments

The table below provides an overview of the functions of the parameter areas and parameter segments.

Parameter area	Function
PAR 00L	Parameter area PAR 00L contains entries concerning text handling and function control; for example, the message header length **for data input** is defined in PAR 00L and PAR 01L.
PAR 00D	Entry of device functions to be executed after completion of data output. Initiation only by the CPU. Bypass mode can be set in character LAP1. The system line can be activated in character AZP.
PAR 10L - PAR 70L	Parameter areas for entry of a print command or of the channel address of a printer terminal connected to the same cluster controller. Initiation by the key corresponding to the parameter area. *Example*: Pressing key LA2 = initiation of function entered in PAR 20L.
PAR 10D - PAR 70D	Structure as for PAR 10L - PAR 70L; used for initiation of LA function by CPU.

Table 4-5 Functions of parameter areas

Parameter segment	Function
PAR 01L	Parameter area PAR 01L contains entries concerning text handling and function control.
PAR 02L	Used only in field mode for field definitions.
PAR 01D - PAR 03D	Entry of text identifiers that are transmitted along with data inputs. The length of the text identifier for input is specified in character TIL of PAR 01L.

Table 4-6 Functions of parameter segments

The coding of the parameter areas and parameter segments is described in [1].

Possible Errors

- A maximum of 1 parameter area with 1, 2, 3 or 4 parameter segments can be loaded with one PAG sequence.
- The termination character must stand at the end of a PAG sequence. If no termination character is present, then when the addressed parameter area/segment has been filled the subsequent characters may overwrite the screen contents.
- If loading of the parameters is carried out by means of a messageheader, the parameters loaded by means of parameter entries are erased.

4.2.1.1 Structure of Parameter Area PAR 00L

The parameter area PAR 00L can be loaded by means both of parameter entries (PAG) and of the PARAMO of the message header.
When a message header is used, the specifications of the PARAMO are taken over into parameter area PAR 00L or PAR 00D of the data display terminal.

Note

Loading by means of the message header is not permitted for new applications.

Character	Short name	Meaning	Implemented in 97.. DDT
1	SAW1	Send command 1	4x, 5x, 6x
2	FST1	Function control character 1 contains the following function assignments: - Insert/delete line - Cursor function (SMF) - Lock/unlock Roll-up function (RU) - Activate/deactivate Roll-up mode (RUB)	
3	FST2	Function control character 1 contains the following function assignments: - "Erase Memory" possible via keyboard - Erase parameter areas with function 'LSP' - Automatic tabulator - Function 'FAZ' - Handling of MOD and BLI bits with LZF or LVD	
4	GEF1	Device function character 1	
5	FST3	Function control character 3 contains the following function assignments: - Control of cursor position after input - Control of cursor position after output - Address of cursor position for data input in old and new mode - Contents of message header - Activate interactive mode - Unlock keyboard	
6	FST4	- Interactive mode or field mode - Cursor also jumps to start of screen in protected fields	
7	GEF2	Insignificant	
	WAR	Queue identifier	

Table 4-7 Overview of characters in PAR 00L

Send Data in Formatted Mode

Depending on the field handling character (FHC) and the send command (SAW), data is transmitted accordingly in formatted mode.

Handling the MOD and BLI Bits

When the 'LZF' function is entered, the MOD and BLI bits - depending on character 3 of PAR 00L (FST2) - are set, reset or remain unchanged.

4.2.1.2 Structure of Parameter Area PAR 01L

The parameter area PAR 01L is used as an extension of the base parameter area PAR 00L and can only be loaded by means of parameter entries (PAG).

Character	Short name	Meaning	Implemented in 97.. DDT
1	TIL	Text identifier length Specifies if and in what length a TI is to be transferred from the mailbox during the next data input.	4x, 5x, 6x
2	TKL	Reserved	
3	FST5	Usage lock for LA keys in the case of activation by the operator and within P functions	
4	FST6	Usage lock for P keys in the case of activation by the operator	
5	FST7	Load lock for P keys P1 through P20	
6	SAW2	Send command SAW2 only for DÜ2	
7	FST8	Request ID card information etc.	
8	—	Reserved	

Table 4-8 Overview of characters in PAR 01L

4.2.1.3 Structure of Parameter Area PAR 00D

The parameter area PAR 00D is loaded through data output from the CPU.
PAR 00D is reserved exclusively for device functions that are to be initiated only when a data output has been completed.
Loading of the parameter area is effected by means of parameter entries (PAG).

Note

> If an LA function is to be initiated by the CPU, then the associated parameter area (PAR 10D - PAR 70D) for the initiated function must be loaded.

Character	Short name	Meaning	Implemented in 97.. DDT
1	GEF1	Device function character 1; possible entries: second character of a 2-character string, announcer for a 3-character string.	4x, 5x, 6x
2	GEF2	Device function character 2 is only interpreted if the announcer for a 3-character string is entered in GEF1.	
3	—	Reserved	
4	LAP1	LA function parameter 1. Data output does/ does not take place on the screen of the data display terminal (bypass).	
5	AZP	Indicator line	
6	—	Reserved	
7	—	Reserved	
8	—	Reserved	

Table 4-9 Overview of characters in PAR 00D

4.2.2 Message Section with Structured Commands

The structured commands are not part of the message header.
The length of a structured command is given in the length field and not in the start-of-message character (`NBZ`); the `NBZ` is `X'40'`.

The message section with structured commands is used for controlling the following functions:

- Read SIDATA Parameters (LSID) (implemented for the 9756, 9758, 9762 and 9763 DDTs)
- Read P-Areas (LPBE) (implemented for the 9756, 9758, 9762 and 9763 DDTs)
- Load Character Generator (LZEG) (implemented for the 9763 DDT)
- Define Screen (DEBS) (implemented for the 9763 DDT)
- Reset Screen (RDSS) (implemented for the 9763 DDT)
- Write Display Area (SRBR) (implemented for the 9763 DDT)
- Query (see section 4.4.1)
- Query Reply (see section 4.4.2) (implemented for the 9758, 9762 and 9763 DDTs)

Error messages provide information on the first error that occurred in a message.

Byte	Meaning	
/	NBZ	Start-of-message character
/	PAF	Parameter field announcer
/		
/		
1	Length	The length, including length field, should be specified as a decimal value with leading zeros.
2		
3		
4		
5	PAFK	Parameter field ID
6		
n		Parameters and/or data

Table 4-10 General structure of structured commands

Note

Structured commands should be sent immediately after the NBZ.

4.2.2.1 Read SIDATA Parameters (LSID)

Implemented for the 9756, 9758, 9762 and 9763 DDTs.

When the read call is received, the DDT sends its SIDATA parameters.

To ensure uniform settings for a group of data display terminals, the SIDATA parameters of a data display terminal selected by the system can be read in so as to allow them to be output to all other DDTs in this group at a suitable time. Manual setting of the DDTs in this group can thus be avoided.

Each halfbyte is provided with a zone portion (X'3x') before transmission in order to exclude data transmission control characters on the transmission path.

4.2.2.2 Read P-Areas (LPBE)

Implemented for the 9756, 9758, 9762 and 9763 DDTs.

P-areas can be loaded manually by the terminal user and by means of the program "PLUS".

The contents of the P-areas can be read by the system through the output of a read call in order to allow subsequent reloading of the P-areas with the same contents at a suitable time.

On receiving the read call, the DDT sends the contents of P-areas P1 through P20.

4.2.2.3 Load Character Generator (LZEG)

Implemented for the 9763 DDT.

The LZEG command allows up to 8 fonts (ZEG0 - ZEG7) to be loaded, depending on the configuration level.

After power-on, only ZEG0 for the standard format 24 x 80 is available. At this time ZEG0 contains the font variant selected with SIDATA.

All other fonts are preset with an "undefined" symbol and must therefore be loaded when required after each power-on.

In the case of a switch in format, a ZEG0 that may have been loaded by the CPU is overwritten again with the symbols for the format concerned, belonging to the standard ZEG.

The LZEG command (Load Character Generator) comprises a parameter segment and a data segment. If only parameters are to be modified, then the data segment can be omitted.

The parameter segment must however be complete, i.e. the parameter segment must comprise 24 (LZEG with continuation), 30 (length of continuation: 6 bytes) or 32 (length of continuation: 8 bytes) bytes.

On receipt of the LZEG command, the functions specified in the parameter segment are executed and the data contained in the data segment is loaded into the specified font, assuming that a data segment was included in the output and that the physical ZEG is loadable. Fonts ZEG1 through ZEG4 are used by the 97801 emulation when the DDT is powered on if the DDT is equipped with a corresponding port. They cannot then be loaded by the CPU (see configuration query reply). The CPU can however also utilize the ZEGs loaded for the 97801 emulation if the fonts are selected with FAT or ZAT.

If a specified ZEG is not loadable, then further processing of the message is aborted and - if requested - an error message is sent to the CPU.

With a single command it is possible to load or erase only one ZEG or one color plane of a triple-plane ZEG. If more than ZEG or color plane is to be loaded, then the command must be repeated with the appropriate parameters.

Each plane of a triple-plane ZEG can be loaded with different contents.

Depending on the parameter entries however, all three planes of a triple-plane ZEG can also be loaded with the same data if a triple-plane ZEG is to be used as a single-plane ZEG.

One message may contain more than one LZEG command. One command may not however be transmitted split over two messages. Using a single command, it is also possible to load contiguous individual segments of a font. Separate commands are required for non-contiguous font segments.

Up to 128 symbols for the codes X'00' through X'7F' can be loaded into each ZEG. For use of the codes X'00' through X'1F', see section 4.2.2.6, Write Display Area.

Note

> In addition to the symbols for the codes X'20' through X'7F', the symbol for the code X'00' must also be loaded for each ZEG required because the functions NUL, LVA, LZF, LVD and LZE enter the code X'00' in the display area.
> If the character box for X'00' is not loaded, then after the functions mentioned above have been used with FAT3 an undefined symbol will be displayed in all erased character positions.

Only one symbolic name may be assigned to a physical ZEG. A symbolic name may be used for only one physical ZEG.

The name of the logical ZEG must be 8 bytes long. The link between the symbolic name and the physical ZEG is established in the parameter segment (byte 16 and bytes 17 through 24).

The name of the ZEG currently in use is displayed in the device indicator line.

The DEZS command is used to switch between fonts (see section 4.2.5.4).

The DEFA command is used to switch colors (see section 4.2.5.4).

In the case of errors in the parameter segment of the command, processing of the message is aborted and - if requested - an error message is sent to the CPU.

Editing of Data for Loading Fonts

Depending on the hardware configuration level of the DDT, 4 or 8 fonts with 128 character boxes each can be loaded for displaying text and graphics. Each **character box** occupies a matrix of 9 x 16 pixels.

Monochrome monitors

The CRT controller for monochrome monitors is equipped with up to 8 fonts (ZEG). Fonts ZEG0 through ZEG7 are designed as single-plane ZEGs. The characters are displayed dark on a light background.

The individual pixels in the matrix can only be switched on or off, depending on their value,

where 0 = off and 1 = on.

Color monitors

The CRT controller for color monitors is equipped with up to 8 fonts. Fonts ZEG0 through ZEG3 are designed as single-plane ZEGs, ZEG4 through ZEG7 as triple-plane ZEGs. With a single-plane ZEG, only the color of the character with a dark background can be specified, while with a triple-plane ZEG a color can be specified for each pixel in the character matrix (9x16).

Single-plane ZEG

With a single-plane ZEG, all pixels of a character box that are switched on have the same color. The color is defined by the character/field attributes. The background, i.e. the pixels not switched on, are dark.

Triple-plane ZEG

If with a triple-plane ZEG all three color planes are loaded with the same data, then all pixels of a character box that are switched on have the same color. The color is defined by the character/field attributes.

If with a triple-plane ZEG all three color planes are loaded with different data, then a pixel that is switched on in one of the three planes is displayed in the color that is defined by the character/field attributes. A pixel will only be dark when it is switched off in all three planes.

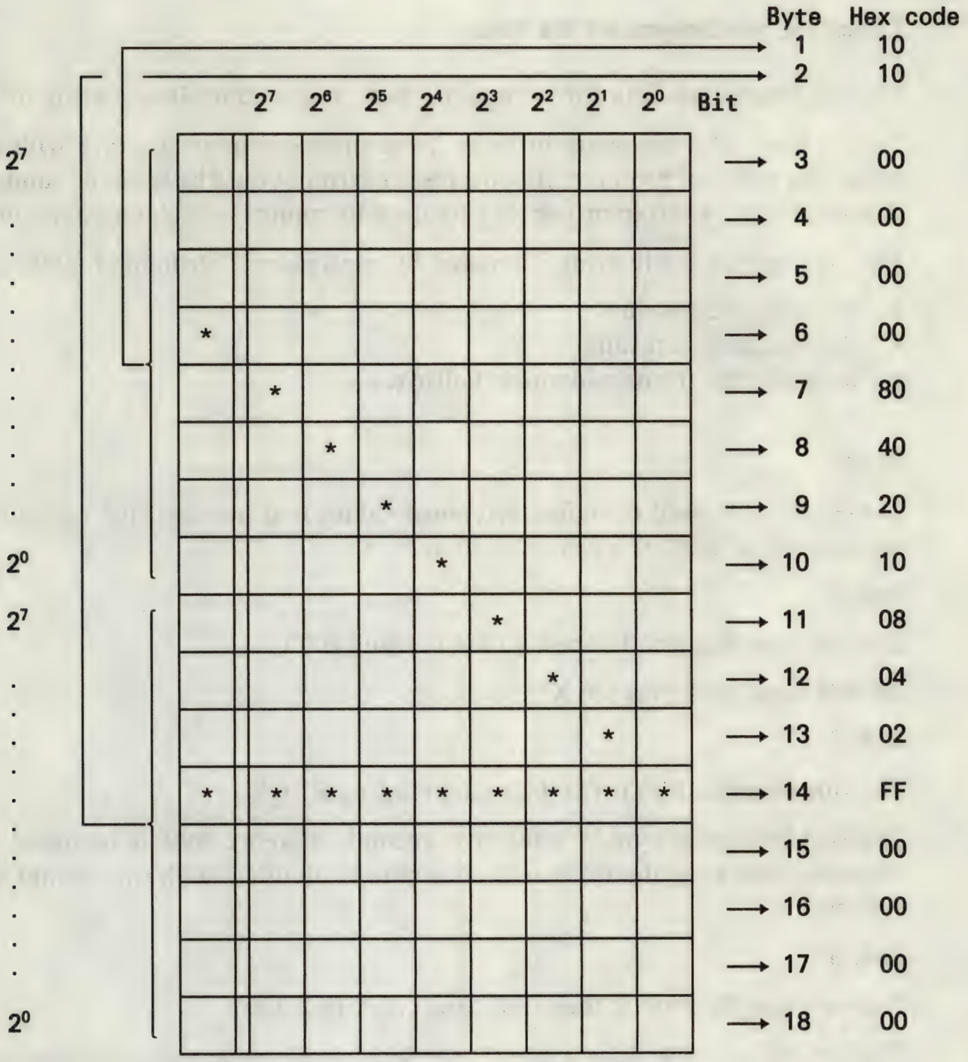
If white is specified as the color attribute in the character/field attributes, then the color of each individual pixel in the character matrix (9 x 16) is produced by additive mixing of the colors from the three color planes. The pixels in the individual color planes can only be switched on or off,

where: 0 = off and 1 = on.

In the data field of the LZEG command, data that is required to fill the character boxes can be transmitted uncompressed or compressed. In uncompressed form 18 bytes are required for each character box; compression allows the number of bytes per character box to be reduced.

Loading Character Boxes without Compression

Each character box occupies a character matrix of 9 x 16 pixels. These 9 x 16 pixels can be represented as follows in hexadecimal code:



* Set pixels

Fig. 4-4 Character representation

As a 7-bit character frame is used for transmission, the hexadecimal code must be converted prior to transmission.

Loading Character Boxes with Compression

The purpose of compression is to produce the shortest possible bit string for loading a character box.

Producing the Compressed Bit String

The uncompressed data for a character box always comprises a string of 18 bytes.

The 18 bytes of a character box can be compressed in accordance with four different rules. The rule that produces the shortest bit string should be used for compression of the character box. A different rule can be used to compress each character box.

The "compressed" bit string is created by contiguously stringing together:

- the rule specification,
- the comparison results,
- if applicable, the non-identical halfbytes.

Rules:

These rules are used to define reference values that are used for comparison with the uncompressed data of a character box.

Rule 1

The rule specification is one bit long (coding B'0').

All reference halfbytes are X'0'.

Rule 2

The rule specification is two bits long (coding B'10').

The first reference byte is X'00'; the second reference byte is identical with the first character box **byte**, the third reference byte is identical with the second character box **byte** etc.

Rule 3

The rule specification is three bits long (coding B'110').

The first reference word is X'00 00'; the second reference word is identical with the first character box **word**, the third reference word is identical with the second character box **word** etc.

Rule 4

The rule specification is four bits long (coding B'1110').

In the character box all bytes are X'00'; no comparison with a reference is required.

Comparison steps

For the comparison, each two successive bytes are combined to form one word; 9 words are therefore to be compressed per character box.

Each four successive bits of a byte are known as a halfbyte. (The low-order halfbyte contains bits 2^0 through 2^3; the high-order halfbyte contains bits 2^4 through 2^7.)

Step 1

The character box word is to be compared with its reference word.

If the word from the character box is identical with the reference word, a "0" bit is entered in the bit string; then step 1 is repeated for the next word. In this case, **no** step 2 must be performed for this word!

If the word from the character box is not identical with the reference word, a "1" bit is entered in the bit string and the procedure continues with step 2.

Step 2

The halfbytes from the character box are to be compared in succession with the corresponding halfbytes of the reference word.

If the halfbyte from the character box is identical with the corresponding halfbyte from the reference word, a "0" bit is entered in the bit string and then the comparison of the next halfbyte is performed.

If the halfbyte from the character box is not identical with the corresponding halfbyte from the reference word, a "1" bit and then the entire non-identical halfbyte from the character box is entered in the bit string; then the comparison of the next halfbyte is performed.

When all the halfbytes of the word have been compared, the procedure continues with step 1 for the following word.

The bit string created in this fashion is provided with a "termination". This termination comprises a minimum of four bits and a maximum of 11 bits in order to pad out the bit string to a byte boundary. All bits in the termination are "1" bits.

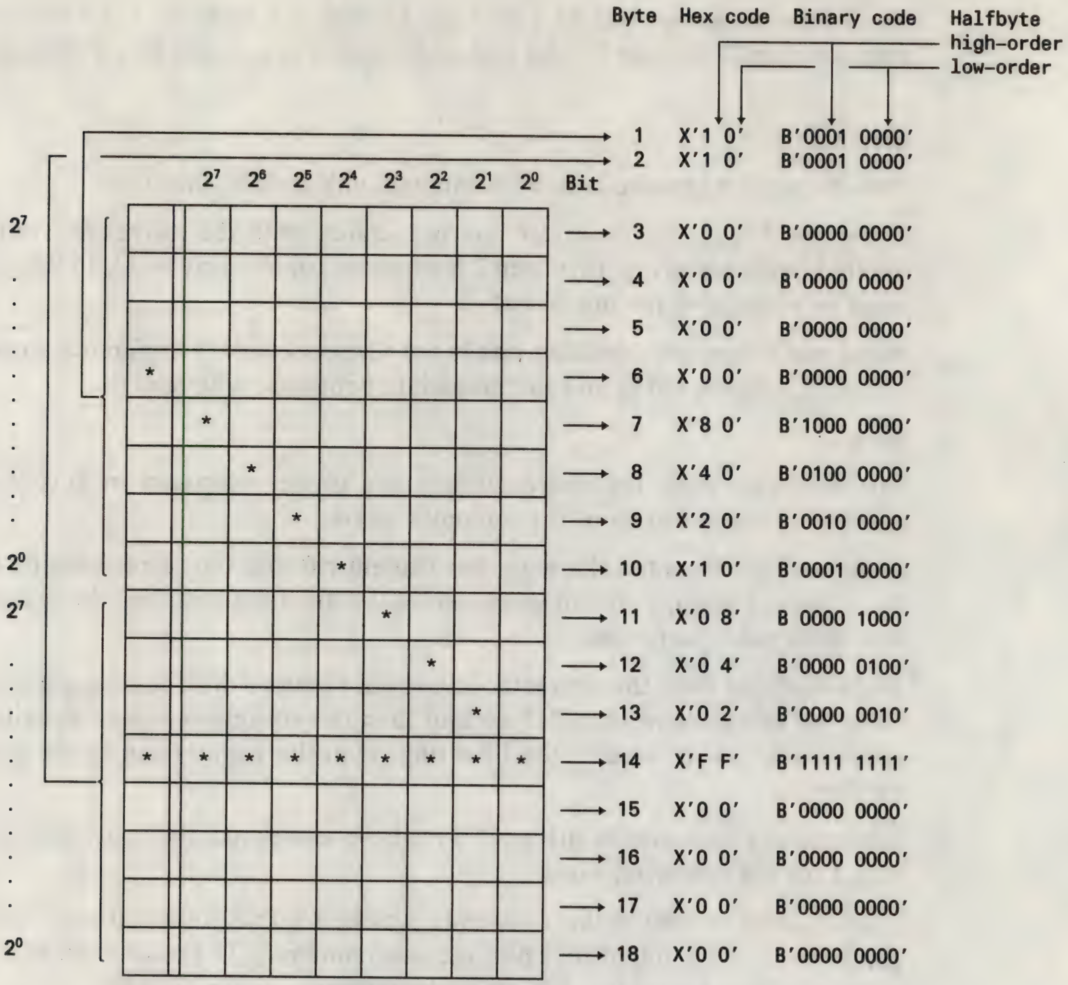
The bit string created thus has the following structure:

Character box 1	Character box 2	...	Character box n	Terminat.
rule,bitstring	rule,bitstring	...	rule,bitstring	111.....1

Fig. 4-5 Structure of created bit string

Example

The character shown below is intended to illustrate the "compression rules".



Byte	Hex code	Binary code
1	X'1 0'	B'0001 0000'
2	X'1 0'	B'0001 0000'
3	X'0 0'	B'0000 0000'
4	X'0 0'	B'0000 0000'
5	X'0 0'	B'0000 0000'
6	X'0 0'	B'0000 0000'
7	X'8 0'	B'1000 0000'
8	X'4 0'	B'0100 0000'
9	X'2 0'	B'0010 0000'
10	X'1 0'	B'0001 0000'
11	X'0 8'	B 0000 1000'
12	X'0 4'	B'0000 0100'
13	X'0 2'	B'0000 0010'
14	X'F F'	B'1111 1111'
15	X'0 0'	B'0000 0000'
16	X'0 0'	B'0000 0000'
17	X'0 0'	B'0000 0000'
18	X'0 0'	B'0000 0000'

* Set pixels

Fig. 4-6 Character representation

The rules

- Rule 1: All reference halfbytes are X'0'.
- Rule 2: The first reference byte is X'00'; the second reference byte is identical with the first character box **byte**, the third reference byte is identical with the second character box **byte** etc.
- Rule 3: The first reference word is X'00 00'; the second reference word is identical with the first character box **word**, the third reference word is identical with the second character box **word** etc.
- Rule 4: In the character box all bytes are X'00'; no comparison with a reference is required. (Not applicable to the example.)

yield the following reference word table for the example:

Reference word			Character box word
Rule 1	Rule 2	Rule 3	
X'0000'	X'0010'	X'0000'	X'1010'
X'0000'	X'1000'	X'1010'	X'0000'
X'0000'	X'0000'	X'0000'	X'0000'
X'0000'	X'0080'	X'0000'	X'8040'
X'0000'	X'4020'	X'8040'	X'2010'
X'0000'	X'1008'	X'2010'	X'0804'
X'0000'	X'0402'	X'0804'	X'02FF'
X'0000'	X'FF00'	X'02FF'	X'0000'
X'0000'	X'0000'	X'0000'	X'0000'

The comparison steps are summarized in tabular form below for each of the three rules (1 - 3) still applicable.

Comparison table with rule 1

Reference word	Character box word	Step 1	Step 2			
			Halfbyte	Halfbyte	Halfbyte	Halfbyte
X'0000'	X'1010'	1	1 0001	0	1 0001	0
X'0000'	X'0000'	0	no step 2			
X'0000'	X'0000'	0	no step 2			
X'0000'	X'8040'	1	1 1000	0	1 0100	0
X'0000'	X'2010'	1	1 0010	0	1 0001	0
X'0000'	X'0804'	1	0	1 1000	0	1 0100
X'0000'	X'02FF'	1	0	1 0010	1 1111	1 1111
X'0000'	X'0000'	0	no step 2			
X'0000'	X'0000'	0	no step 2			

Table 4-11 Comparison table with rule 1

Use of rule 1 produces the following bit string:

0110 0010 1000 1000 1110 0001 0100 0110 0100 1000 1010 1100 0010 1001 0100 1011 1111 1111 00

Interpretation of the bit string:

- The leading "0" is the coding for rule 1.
- This is followed by the bits given in the table (line by line).

With this example, compression of a string of 144 bits produces a string of 74 bits.

Comparison table with rule 2

Reference word	Character box word	Step 1	Step 2			
			Halfbyte	Halfbyte	Halfbyte	Halfbyte
X'0010'	X'1010'	1	1 0001	0	0	0
X'1000'	X'0000'	1	1 0000	0	0	0
X'0000'	X'0000'	0				
X'0080'	X'8040'	1	1 1000	0	1 0100	0
X'4020'	X'2010'	1	1 0010	0	1 0001	0
X'1008'	X'0804'	1	1 0000	1 1000	0	1 0100
X'0402'	X'02FF'	1	0	1 0010	1 1111	1 1111
X'FF00'	X'0000'	1	1 0000	1 0000	0	0
X'0000'	X'0000'	0				

Table 4-12 Comparison table with rule 2

Use of rule 2 produces the following bit string:

1011 0001 0001 1000 0000 0111 0000 1010 0011 0010 0100 0101 1000 0110 0001 0100 1010 0101 1111 1111 1110 0001 0000 000

Interpretation of the bit string:

- The leading bit string "10" is the coding for rule 2.
- This is followed by the bits given in the table (line by line).

With this example, compression of a string of 144 bits produces a string of 95 bits.

Comparison table with rule 3

Refer- ence word	Character box word	Step 1	Step 2			
			Halfbyte	Halfbyte	Halfbyte	Halfbyte
X'0000'	X'1010'	1	1 0001	0	1 0001	0
X'1010'	X'0000'	1	1 0000	0	1 0000	0
X'0000'	X'0000'	0				
X'0000'	X'8040'	1	1 1000	0	1 0100	0
X'8040'	X'2010'	1	1 0010	0	1 0001	0
X'2010'	X'0804'	1	1 0000	1 1000	1 0000	1 0100
X'0804'	X'02FF'	1	0	1 0010	1 1111	1 1111
X'02FF'	X'0000'	1	0	1 0000	1 0000	1 0000
X'0000'	X'0000'	0				

Table 4-13 Comparison table with rule 3

Use of rule 3 produces the following bit string:

1101 1000 1010 0010 1100 0001 0000 0011 1000 0101 0001 1001 0010 0010 1100 0011 0001 0000 1010 0101 0010 1111 1111 1110 1000 0100 0100 00

Interpretation of the bit string:

- The leading bit string "110" is the coding for rule 3.
- This is followed by the bits given in the table (line by line).

With this example, compression of a string of 144 bits produces a string of 110 bits.

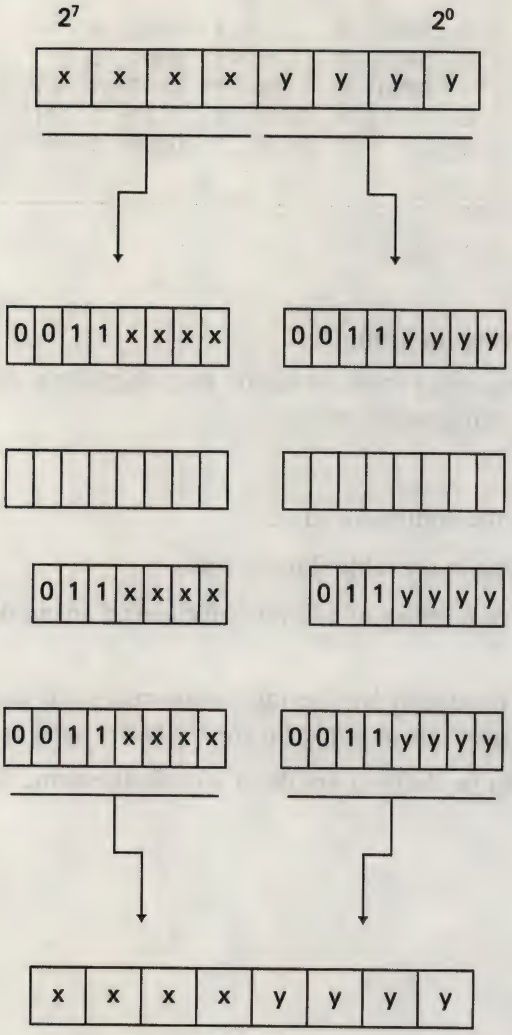
If the lengths of the bit strings produced by the individual rules are compared, it is apparent that rule 1 is the appropriate one to apply for the character used in the example.

The further characters in a font to be defined are dealt with in the same fashion.

Coding of Data in Data Field for Data Transmission

As a 7-bit character frame is used for transmission, the hexadecimal code must be converted prior to data transmission.

The hex code from a character matrix and in graphic mode may assume all code combinations between X'00' and X'FF'.



Hex code

Splitting the 8-bit hex code into two halfbytes, each of which is prefixed by a zone portion X'3', produces a transmittable 7-bit code.

Conversion of characters (ISO 7-bit code) into EBCDIC. 1)

Conversion from EBCDIC into ISO 7-bit code for data transmission. 2)

After removal of the X'3' zone portions, the original 8-bit hex code is reconstituted in the DDT.

Hex code

1) This code conversion is required when the data is passed via system components that in their turn perform a conversion from EBCDIC into ISO 7-bit code.

2) Code conversion as performed with PDN and other system components, depending on their settings.

Fig. 4-7 Coding of data in data field for data transmission

In the DDT, the 7-bit code is converted back into 8-bit hex code and entered in the character matrix of the character.

4.2.2.4 Define Screen (DEBS)

Implemented for the 9763 DDT.

The DEBS command is used to set the screen to the specified format.

The contents of the active display area are always displayed on the screen.

The screen is converted to the specified physical format.

The following formats are possible:

Format	Columns/lines	No. of chars.	Char. matrix x / y	Character area x / y
F1	80 x 24	1920	9 x 16	7 x 12
F2				1)
F3				1)
F4	132 x 27	3564	6 x 14	5 x 10

1) Format not defined at time of going to print.

Fig. 4-8 Screen formats

Display area 0 is set up in the specified format and displayed on the screen.

The contents of the display area, including the planes of the character and field attributes, are erased.

The planes for the character/field attributes and all defined font assignments (LOGZEG to PHYZEG) are however retained.

The font (national or special customer-specific variant) belonging to this format is loaded into ZEG0.

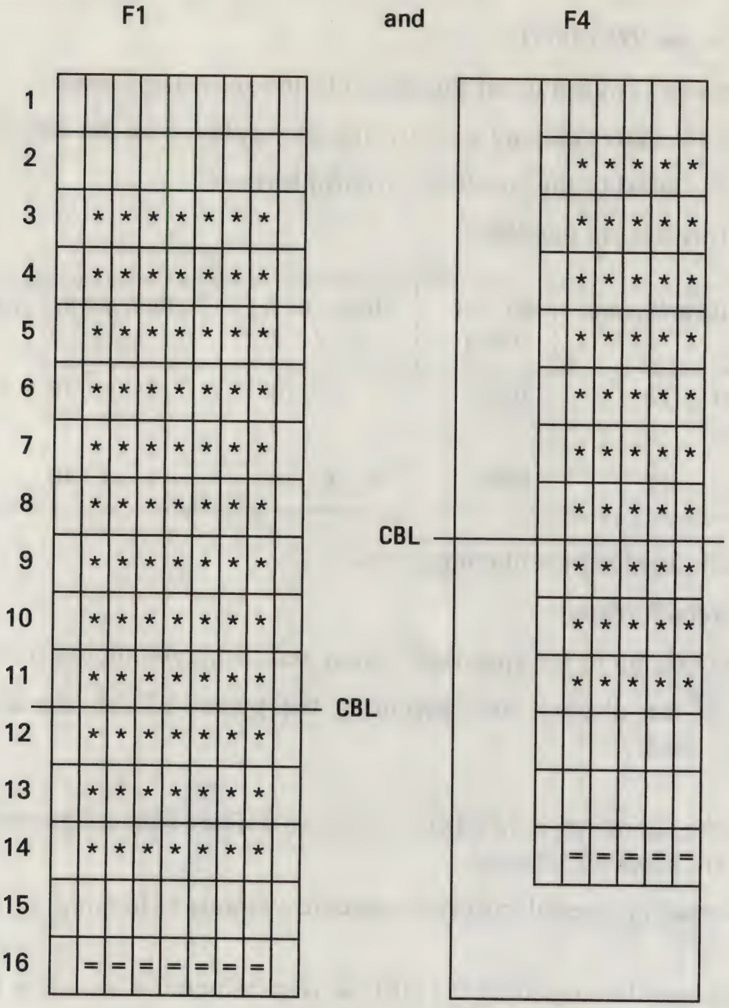
The other fonts must be loaded by the CPU as required after a switch in format; a new font assignment must also be made in such cases.

The character field matrix cannot be modified.

With a color monitor, the characters are displayed colored on a dark background.

In the case of errors in the parameter segment of the command, processing of the message is aborted and - if requested - an error message is sent to the CPU.

0.0.0.0 Character Boxes for Screen Formats



* * * * *	Character area in the standard fonts
CBL	Character base line
= = = =	Position of cursor as underscore

Fig. 4-9 Character boxes for screen formats

Note:

F2 and F3 not defined at time of going to print.

4.2.2.5 Reset Screen (RDSS)

Implemented for the 9763 DDT.

The RDSS command is used to reset the DDT to a defined status.

- The contents of the display areas and of the planes for fields and character attributes are erased.
- The display area is set up in format F1 (24 x 80) and displayed on the screen.
- With a color monitor, the colors are displayed green on a dark background.
- The font (national or special customer-specific variant) belonging to format F1 is loaded into ZEG0.

  The other fonts must be loaded by the CPU as required after the RDSS command; a new font assignment must also be made in such cases.

In the case of errors in the parameter segment of the command, processing of the message is aborted and - if requested - an error message is sent to the CPU.

4.2.2.6 Write Display Area (SRBR)

Implemented for the 9763 DDT.

Transmission of messages between the CPU and a DDT is handled using a 7-bit character frame in the 'non-transparent mode' of the MSV1 procedure or the 'BAM' procedure. Some characters with codes between X'00' and X'1F' are used as data transmission control characters by these procedures.

The data field of the SRBR command can contain data in 'non-expanded mode' or in 'expanded mode' for the specified display area.

In non-expanded mode, only data with a restricted subset of the ISO 7-bit code (no transmission control characters) can be transmitted.

With expanded mode, each data character to be transmitted must be split into two characters, each having a 'zone portion' (X'3x'). In this way it is also possible to transmit characters having codes between X'00' and X'1F' if the code subset restricted by the transmission control characters does not suffice for special applications.

One message can contain more than one SRBR command.

If data is to be transmitted in expanded mode and non-expanded mode in a single message, this data must be transmitted with separate commands.

In the case of errors in the parameter segment of the command, processing of the message is aborted and - if requested - an error message is sent to the CPU.

Note

> This command is required only for characters in the code range X'01' through X'1F'. Characters in the code range X'20' through X'7F' should be transmitted without use of this command.

Data Transmission to DDT

All characters are entered in the specified display area with their character attributes; the relevant preset values are used for these character attributes.

If no character attributes have yet been defined, the default values are entered as the character attributes.

The data segment of the SRBR command may not contain any transmission control characters in expanded mode.

Data Transmission to CPU

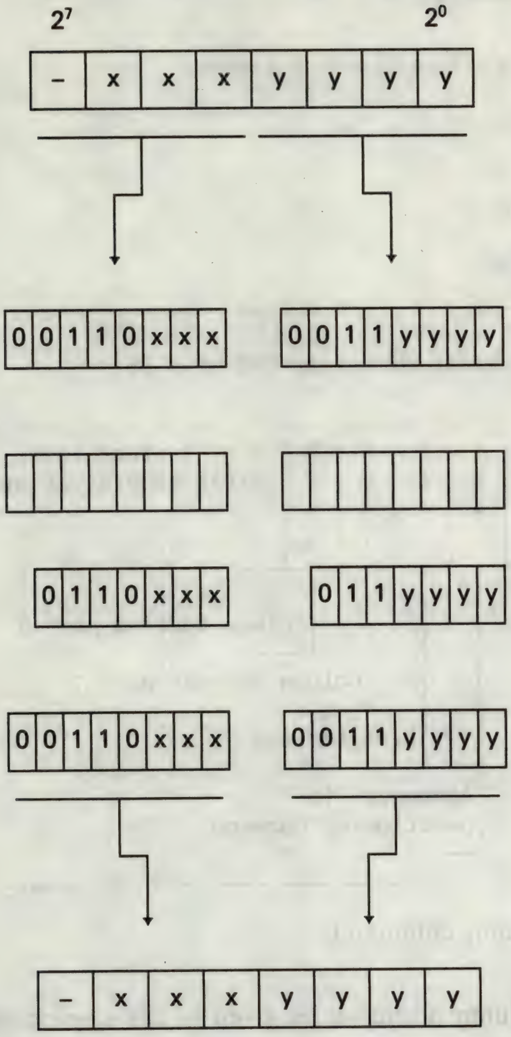
Characters with the codes X'00' through X'7F' where ZAT = default value are transmitted as previously without code conversion to the CPU (for compatibility with previous DDTs).

When ZAT = ZEG0, the codes X'06' (AM), X'19' (EM) and X'1C' (LZE) in the display area are not treated as characters during data transfer to the CPU. They mark the current write position on initiation of the respective function by means of symbols from ZEG0. The corresponding ESC sequences for AM and LZE (for AM: X'1B' X'20' X'50' and for LZE: X'1B' X'71') are transmitted to the CPU instead of these 'markings'. The EM is transmitted only when SAW = X'43'.

Characters with the codes X'00' through X'1F' where ZAT = ZEG1 through ZEG7 are transmitted in expanded mode and thus by means of the SRBR command.

Coding for Transmission with Expanded Mode

Each two characters from the data field that are received are combined to form one character and passed to the display area.



Hex code

Splitting the 8-bit hex code into two halfbytes, each of which is prefixed by a zone portion X'3', produces a transmittable 7-bit code.

Conversion of characters (ISO 7-bit code) into EBCDIC. 1)

Conversion from EBCDIC into ISO 7-bit code for data transmission. 2)

After removal of the X'3' zone portions, the original 8-bit hex code can be reconstituted in the CPU.

Hex code

1) This code conversion is required when the data is passed via system components that in their turn perform a conversion from EBCDIC into ISO 7-bit code.

2) Code conversion as performed with PDN and other system components, depending on their settings.

Fig. 4-10 Coding of data in data field for data transmission

4.2.2.7 Error Message for Structured Commands

The first error detected in a message is specified in the error message by the CPU. Any subsequent errors cannot be reported as the error stack can accommodate only one error. Subsequent errored commands are ignored.

In addition, the code for the command and the error code are displayed on the screen.

4.2.3 Message Section with Positioning Commands (IS4 Sequence)

Positioning commands within a message section make it possible to access any write position on the data display terminal with the cursor.

Any number of positioning commands may be contained in any order in the message.

Structure of Positioning Commands

Positioning commands must always comprise 4 characters,

- the announcer (`IS4 = X'1C'`),
- the line address (`ZLA`),
- the column address part 1 (`SPA1`),
- the column address part 2 (`SPA2`),

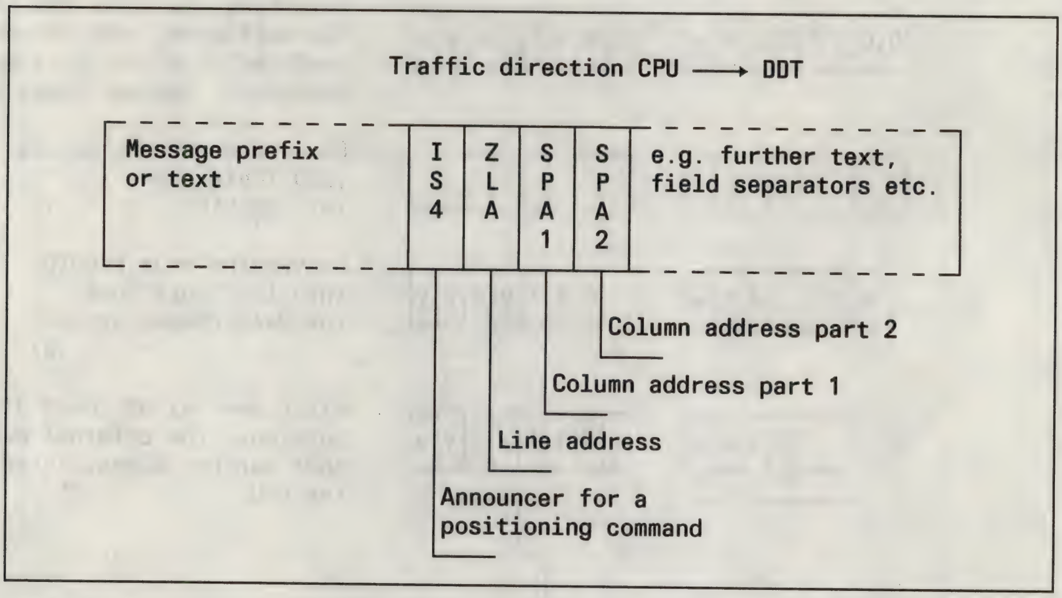

Traffic direction CPU ⟶ DDT
Message prefix or text
IS4
ZLA
SPA1
SPA2
e.g. further text, field separators etc.
Column address part 2
Column address part 1
Line address
Announcer for a positioning command

Fig. 4-11 Structure of positioning commands

The coding for the line and column addresses is given in the appropriate tables in manual [1].

If the cursor is to be positioned to columns 81 through 132, then X'2F' must be specified in SPA1 and the appropriate column must be specified in SPA2.

In the case of positioning operations to columns 1 through 80, SPA2 will contain X'30'.

4.2.4 Message Section with Field Separators and Attributes (IS2, IS3 Sequences)

Note

> For the variants of the 9763 family of DDTs, see also section 4.2.5, Message Section with Field and Character Attributes.

Field separators (FTZ) define the characteristics of a field.

A field begins with at least one field separator (FHC/DCC pair) and ends at the next start of field or at start of screen.

A field must be at least 1 character long. A maximum of one FHC/DCC pair may be used per character.

The following are designated as field separators:

- **Field handling characters** (FHC); these define the field handling characteristics of the characters in a field up to the next FHC (e.g. protected, numeric, printable).
- **Display control characters** (DCC); these define the display characteristics of the characters in a field up to the next DCC (e.g. normal, low intensity, flashing).

Both field separators must always be used together. They are entered in the dummy locations present for each character.

Exception:
When a system message is output, no FHCs may be used!

Field separators

- are announced by means of announcer characters (code extension characters) IS2 and IS3,
- always stand at the beginning of a field,
- are entered in the memory of the data display terminal only by means of a data output,
- remain invisible between the visible characters on the screen,
- do not restrict the number of displayable characters per line,
- cannot be entered via the keyboard.

Structure of Character String with Field Separators

The character string for constructing a message section with field separators comprises

- the announcer (code extension character) IS2 (X'1E') for an FHC,
- the field handling character (FHC),
- the announcer (code extension character) IS3 (X'1D') for a DCC,
- the display control character (DCC).

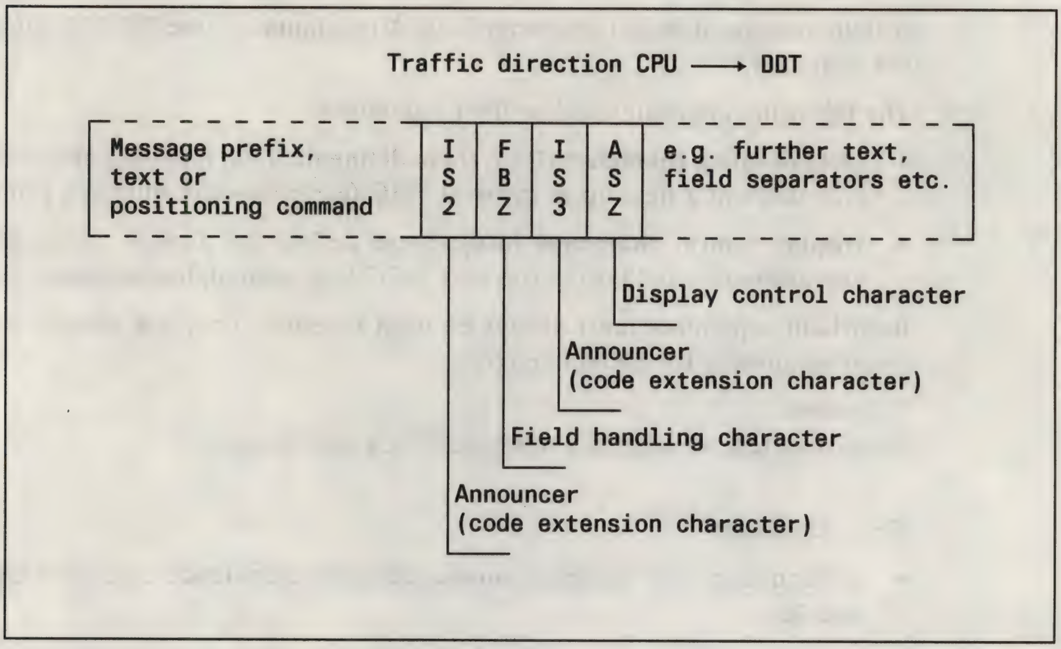

Traffic direction CPU ⟶ DDT
Message prefix, text or positioning command
I S 2
F B Z
I S 3
A S Z
e.g. further text, field separators etc.
Display control character
Announcer (code extension character)
Field handling character
Announcer (code extension character)

Fig. 4-12 Character string with field separators

4.2.4.1 Entering Field Separators

If the dummy locations of a character are not yet occupied, then the field separator received is entered in them.

4.2.4.2 Modifying Field Separators

The field separator contained in dummy locations is replaced by the field separator received.

The modification is effected by means of a data output from the CPU.

4.2.4.3 Erasing Field Separators

Field separators can be erased by means of the following "Erase command":

IS2 NUL IS3 NUL (X'1E001D00')

If the dummy locations contains no field separator, then the "Erase command" will be ignored.

4.2.5 Message Section with Field and Character Attributes

The variants of the 9763 family of DDTs can process **extended** field and character attributes.

The extended field and character attributes allow:

- definition of the old field and character attributes,
- definition of fonts,
- definition of colors,

where:

- field attributes (FAT) apply from definition up to the next FAT or to end of screen,
- character attributes (ZAT) are assigned to the individual characters.

To facilitate more flexible handling of these attributes, control is implemented by means of **structured commands.**

- The fonts are loaded by means of the structured command LZEG (see section 4.2.2.3).
- The structured command SRBR is used for writing the logical display area (see section 4.2.2.6).
- The structured command DEBS allows the physical format of the screen to be switched (see section 4.2.2.4).
- Character and field attributes are created and modified using the commands EZFA, MOFA, RSZA, DEFA and DEZS described below.

The attributes are written to separate working storage planes. These planes are set up when the DDT is powered on.

Each field begins with the four field attributes FAT1 through FAT4.

The **field attributes FAT1 and FAT4** assume the functions of the **field separators; FAT2, FAT3** and the character attributes are redefined.

The character attributes comprise the display attribute (ZAT1), color and font selection (ZAT2 and ZAT3) (see section 4.2.5.4).

The individual character attributes ZAT1, ZAT2 and ZAT3 are independent of one another, i.e. each character attribute can be modified separately.

4.2.5.1 Field Attributes (FAT)

The field attributes can be created and modified only by the CPU.

Exception
Automatic "modification" of a field attribute in the case of character input using the keyboard.

The characteristics defined in the field attributes apply generally to all characters in this field up to the next field attribute.

However, additional character attributes may also be defined for each character (see section 4.2.5.4 Character Attributes).

If there is no field attribute at start of screen, a dummy attribute with default values applies to the characters up to the first field attribute.

The last field ends at end of screen.

The number of fields is limited by the number of characters per line.

If the planes for the field attributes FAT2 and FAT3 and the character attributes are not created, the commands EZFA and MOFA are ignored. An error message will be issued, if requested (PAR01L-FST8-AFEM = 1).

Field Attribute 1

Field attribute 1 (FAT1) assumes the function of the display control character (DCC).

FAT1 defines the display characteristics for the characters of a field up to the next FAT1.

When 'Characters invisible' (ZNS) is specified in FAT1 or in ZAT1, the characters are not visible.

Default value: BLI, UNT, ZNS, HAH and INV = null.

On DDTs with color monitors, colors are assigned to the display characteristics Flashing, Underscored and Low intensity (including combinations of these characteristics) since these display characteristics cannot be implemented.

In the case of color monitors, certain colors can be assigned to some display characteristics of a FAT1 by means of SIDATA tables.

In order to avoid the creation of data transmission control characters, bit 7 must = 1 for data transfer.

The following display characteristics can be specified:

Function	Display characteristics
(none)	
(none)	
(none)	
INV	Inverse: Monochrome monitor: When 'Characters dark, background light' is set, the characters in these fields are displayed light on a dark background. Color monitor: Dark character and colored background dependent on SIDATA setting if no FAT2 or ZAT2 present.
HAH	Low intensity: Monochrome monitor: Characters are displayed at reduced brightness. Color monitor: Colored character dependent on SIDATA setting if no FAT2 or ZAT2 present.
ZNS	Characters invisible: The characters in these fields are invisible. (Example of use: invisible password input)
UNT	Underscored: Monochrome monitor: The characters are displayed with underscoring. Color monitor: Colored character dependent on SIDATA setting if no FAT2 or ZAT2 present.
BLI	Flashing: The BLI bit in the FAT1 is set when a markable field is marked by the DDT. Re-marking, and CPU outputs that affect the display memory, cause the BLI bit to be reset in marked fields. Monochrome monitor: The characters are displayed flashing. Color monitor: Colored character dependent on SIDATA setting if no FAT2 or ZAT2 present.

Table 4-14 Display characteristics in FAT1

Combinations of the individual characteristics are possible.

Field Attribute 2

Field attribute 2 (FAT2) specifies which color is to be used for displaying this field. If no field attribute FAT2 is specified, or if no color is specified in FAT2 (FAT2 = X'00' = default value), the characters are then displayed in the color according to FAT1.

Field Attribute 3

Field attribute 3 (FAT3) specifies which font is to be used for displaying the characters in this field. If no field attribute FAT3 is specified, or if no font is specified in FAT3 (FAT3 = X'00' = default value), the characters are displayed with symbols from ZEG0.

Field Attribute 4

Field attribute 4 (FAT4) assumes the functions of the field handling character (FHC).

The FAT4 defines the characteristics of a field up to the next FAT4, i.e. one field ends only when the next field begins. The characteristics defined in FAT4 apply to all characters in this field regardless of the character attributes.

If no field attribute FAT4 is specified, the characters are then displayed with the characteristics: GNU, NUM, MOD, MAR and GÜ = 0, DRU = 1.

In order to avoid the creation of data transmission control characters, bit 7 must = 1 for data transfer.

The following field characteristics can be specified:

Function	Field characteristics
(none)	
(none)	
GÜ	Protected, transmittable: The DDT user cannot change the contents of the fields by entering characters. The contents of these fields are however transmitted to the CPU.
DRU	Printable: These fields can be printed by a printer.
MAR	Markable: The field characteristic "Markable" is effective only in conjunction with a FAT1. Markable fields can be marked by means of the MAR key. Marking of a markable field sets the MOD bit in the FAT4 and the BLI bit in the FAT1. Re-marking in protected markable fields causes the MOD bit in the FAT4 and the BLI bit in the FAT1 to be reset; in unprotected fields only the BLI bit in the FAT1 is reset.
MOD	Modified: The MOD bit in the FAT4 of a field is set by input of valid characters, by insertion or deletion of characters, by marking and by erase functions (see device functions). The MOD bit can also be predefined by the CPU The MOD bit is reset: . by the character string ESC FAZ (X'1B6D' = ESC m). . automatically after each data output, depending on entries in PAROOL-FST2-FAZ. . by functions LZF and LVD (see PAROOL-FST2). . by re-marking protected markable fields.
NUM	Numeric: In numeric fields, only input of the codes X'2A' through X'39' is possible via the keyboard. In the ZEGO these are the digits and the special symbols * + , - . / as per DIN 66003.
GNU	Protected, non-transmittable: The terminal user cannot change the contents of the fields by entering characters. The contents of these fields are not transmitted to the CPU.

Table 4-15 Display characteristics in FAT4

Combinations of the individual characteristics are possible.

Transmission of Field Attributes to CPU

Field attributes are transmitted to the CPU only by means of the send command SAW = X'43'.

Field attributes FAT1 through FAT4 are combined here to form an EZFA command (for structure, see section 4.2.5.2 Create Field Attributes (EZFA)).

If the default value is entered for one of the field attributes FAT1 through FAT4 in the display area, then this field attribute (including specification of the plane) is not transmitted.

If default values are entered for all field attributes, then the EZFA command comprises only the three characters (X'1D2930').

The 'old' field separators (FHC and DCC) are transmitted in the form previously specified.

4.2.5.2 Create Field Attributes (EZFA)

The EZFA command is used to set up a field at the current write position, having the particular field attributes specified.

Any field attributes that may already exist and also old FHCs and DCCs are erased in the process.

All four field attributes are always created.

If a field attribute (FAT1 through FAT4) is not specified in the EZFA command, or if the value X'00' is entered, then a 'default value' is used in the relevant plane for this field attribute.

Each attribute comprises a pair of characters (plane and attribute).

The order of the attributes in the command is optional.

The characters and any existing character attributes are not changed by creation of the field attributes.

All characters with default values in the character attributes (see section 4.2.5.4) assume the characteristics defined for the field in the corresponding field attributes.

The preset values for the character attributes (ZAT1, ZAT2 and ZAT3) are erased upon receipt of an EZFA command, i.e. for all characters that follow this command the default values are used as character attributes until new character attributes are specified.

In the case of errors in the parameter segment of the command, processing of the message is aborted and - if requested - an error message is sent to the CPU.

If the specified font (with a triple-plane, all three planes) is not loaded, or if no physical font has been assigned to the specified logical font, then the relevant default values are entered as character attributes for the characters following.

4.2.5.3 Modify Field Attributes (MOFA)

The MOFA command can be used to modify the characteristics of a field.
Only those field attributes specified in the command are modified. The field attributes for the non-specified planes are not modified.

The new field attributes specified in the command become effective.

If X'00' is entered in ANZ in the MOFA command, then all FATs (FAT1 through FAT4) at this position are eliminated.

The command is executed only if the cursor is positioned at the first character of a field.

If there is no field attribute at the position, an error message will be issued if this has been requested (PAR01L-FST8-AFEM = 1).

The preset values for the character attributes are retained unchanged.

In the case of errors in the parameter segment of the command, processing of the message is aborted and - if requested - an error message is sent to the CPU.

If the specified font (with a triple-plane, all three planes) is not loaded, or if no physical font has been assigned to the specified logical font, then the relevant default values are entered as character attributes for the following characters.

4.2.5.4 Character Attributes

The character attributes comprise the display attribute (ZAT1), color and font selection (ZAT2 and ZAT3).

The individual character attributes ZAT1, ZAT2 and ZAT3 are independent of one another, i.e. each character attribute can be modified separately.

Character attributes are always tied to the characters. When a character is erased, all character attributes relating to this character are then also erased.

If characters change their position (due to Delete or Insert Character, Delete or Insert Line, Roll-up), then the character attributes for these characters also change their position.

Character attributes for a character can be modified by the CPU only through the output of new characters with different character attributes.
The character attribute for a character that has already been input cannot subsequently be modified.

When a character is entered, then the character attributes for this character are also entered in the relevant attribute planes.

The character attributes ZAT1, ZAT2 and ZAT3 are taken here from 'preset values'. Where no character attributes have been defined, the respective presets for ZAT1, ZAT2 and ZAT3 contain the default value X'00'. The predefinitions from the corresponding field attributes are then used for displaying the characters.

The preset values for the character attributes are defined by means of the DEAA, DEFA and DEZS commands.

The relevant default values are entered as presets by means of the RSZA and EZFA commands.

If the character attributes for the characters are erased, then the respective default values for the character attributes are used for displaying the characters.

The default values for the character attributes are the corresponding field attributes.

The WDH function overwrites all characters including the character attributes with the specified character and the current presets up to the specified address without modifying the presets.

The NUL function erases an unprotected character, including its character attributes, at the current write position without modifying the presets.

The LVA function erases all characters, including their character attributes, from unprotected fields up to the specified address without modifying the presets.

The LZF function erases all unprotected characters, including their character attributes, from the cursor to the end marker, end of field or end of line without modifying the presets.

The LVD function erases all unprotected characters, including their character attributes, from start of screen or from the cursor to the end marker or end of screen without modifying the presets.

The LZE function erases all characters, including their character attributes, up to end of field or end of line without modifying the presets.

Transmission of Character Attributes to DDT

If an output begins without specification of character attributes, then default values (ZAT1 = X'00', ZAT2 = X'00' and ZAT3 = X'00') are entered as character attributes for these characters until character attributes are specified.

Transmission of Character Attributes to CPU

For input, the character attributes are transmitted to the CPU in the same form as for output (see DEAA, DEFA and DEZS commands).

Prior to the first character of a field created by means of the EZFA command, character attributes are transmitted only when no default values are used as character attributes.

Prior to subsequent characters in this field, character attributes are transmitted only when the character attributes have changed within the field.
In the case of a number of successive characters having the same character attributes (ZAT1, ZAT2 and ZAT3), specification of the character attributes is made only before the first character of this string.
If the character attributes change individually, then only the modified character attribute is transmitted.

On switching to default values within a field, the relevant default values are transmitted as character attributes, i.e. X'00'.

When ZAT = ZEG0, the codes X'06' (AM), X'19' (EM) and X'1C' (LZE) in the display area are not treated as characters during data transfer to the CPU. They mark the current write position on initiation of the respective function by means of symbols from the ZEG0.
The corresponding ESC sequences for AM and LZE (for AM: X'1B' X'20' X'50' and for LZE: X'1B' X'71') are transmitted to the CPU instead of these 'markings'. The EM is transmitted only when SAW = X'43'.

Reset Character Attribute (RSZA)

The RSZA command sets the presets for the character attributes ZAT1, ZAT2 and ZAT3 to default values.

Default values for

- ZAT1 = FAT1
- ZAT2 = FAT2
- ZAT3 = FAT3

All characters following this command in the same message are displayed with the relevant default values. Characters from other messages remain unmodified.

The RSZA command may not be used until planes have been set up for character attributes.

Define Display Attribute (DEAA)

This display characteristic is always entered for each character.

A switch in the display characteristic is made by means of the DEAA command. The command must precede the first character in the message that is to be displayed with this display characteristic. The command acts as a preset for ZAT1. This preset is modified by means of a new DEAA command.

Default value for ZAT1 is FAT1.

All characters following this command in the same message are displayed with the specified display characteristics. Characters from other messages remain unmodified.

When 'Characters invisible' is specified in FAT1 or in ZAT1, the characters are not visible.

Define Color Attributes (DEFA)

When bit 3 = 1 in FAT1, the characters are invisible, i.e. characters and background are displayed in the same color.

The color attribute is always entered for each character.

A switch in the color attributes is made by means of the DEFA command. The command must precede the first character in the message that is to be displayed with these color attributes. The command acts as a preset for ZAT2. This preset is modified by means of a new DEFA command.

Default value for ZAT2 is FAT2.

All characters following this command in the same message are displayed in the specified color. Characters from other messages do not change their color.

When a triple-plane ZEG is used, for each character that is to be displayed colored (with a color attribute), the triple-plane ZEG must have been loaded beforehand.

Define Font (DEZS)

The font is always entered for each character.

A switch in font is made by means of the DEZS command. The command must precede the first character in the message that is to be displayed with this font. The command acts as a preset for ZAT3.
This preset is modified by means of a new DEZS command.

Default value for ZAT3 is FAT3.

The character attribute for a character that has already been input cannot subsequently be modified.

All characters following this command in the same message are displayed using symbols from the specified font. Characters from other messages remain unchanged.

It is possible to switch from any font to any other desired font.

This does not exclude the use of a ZEG loaded for the 97801 emulation.

For each character that is used in a font (including Space etc.), the character matrix must also be loaded by means of the LZEG command.

If no physical ZEG is present or if no logical ZEG was assigned to a physical ZEG, the command will be ignored.

4.2.6 Message Section with Device Functions (ESC Sequences)

A wide range of device functions gives the programmer freedom to structure the screen contents and facilitates the user's operations at the data display terminal.

The following device functions can be executed:

- cursor positioning functions,
- text manipulation functions,
- erase functions,
- data transmission functions,
- functions for system-internal data exchange,
- programmable key functions,
- special functions.

Device functions can be entered in the

- message prefix,
  - through specification in the message header (prohibited for new applications),
- message section,
  - through specification in parameter entries (`PAG`),
  - through specification in the message text itself.

In the message text, device functions are announced by means of announcer characters (code extension characters):

- `ESC` for a 2-character string,
- `ESC` and `Space` for a 3-character string.

In the message header and in parameter entries, the announcer `ESC` is not used.

Restriction:

Only 2-character strings can be entered in a message header.

2-character strings and 3-character strings (`GEF1` and `GEF2`) can be entered in `PAR 00D`.

The device functions `LVA` and `WDH` can be entered only in the text segment.

Device control characters for

- data transmission functions (`Düx`, `Kx`, `Fx`),
- system-internal data exchange (`LAx`) and
- programmable key functions, in which one of the aforementioned functions is contained,

can be entered only in the message header or in `PAR 00D`.

Should these functions nevertheless be entered in the text segment as an `ESC` sequence, they will be ignored because the functions require an end address. The data display terminal also sets an error message in character 6 (`FEM`) of `PAR 01E`.

Execution of device functions:

For data output, entry of the device functions is made in PAR 00D. The function concerned is not executed until transmission of the message has been completed. When the device functions are entered as an ESC sequence within the text segment of a message section, the function concerned is executed immediately - during the current text display on the screen.
For data input from a data display terminal, the code for the data transmission key used (Düx, Kx, Fx) is entered in character 7 (CDS) of PAR 00E.

Structure of Character String for Device Functions in Text

The character string comprises

- the announcer (code extension character) for a 2-character string:
  ESC (X'1B') or
- the announcer (code extension character) for a 3-character string:
  ESC-Space (X'1B20')
- the device function.

Exception

The functions LVA and WDH require end addresses. These must immediately follow the "device function character".

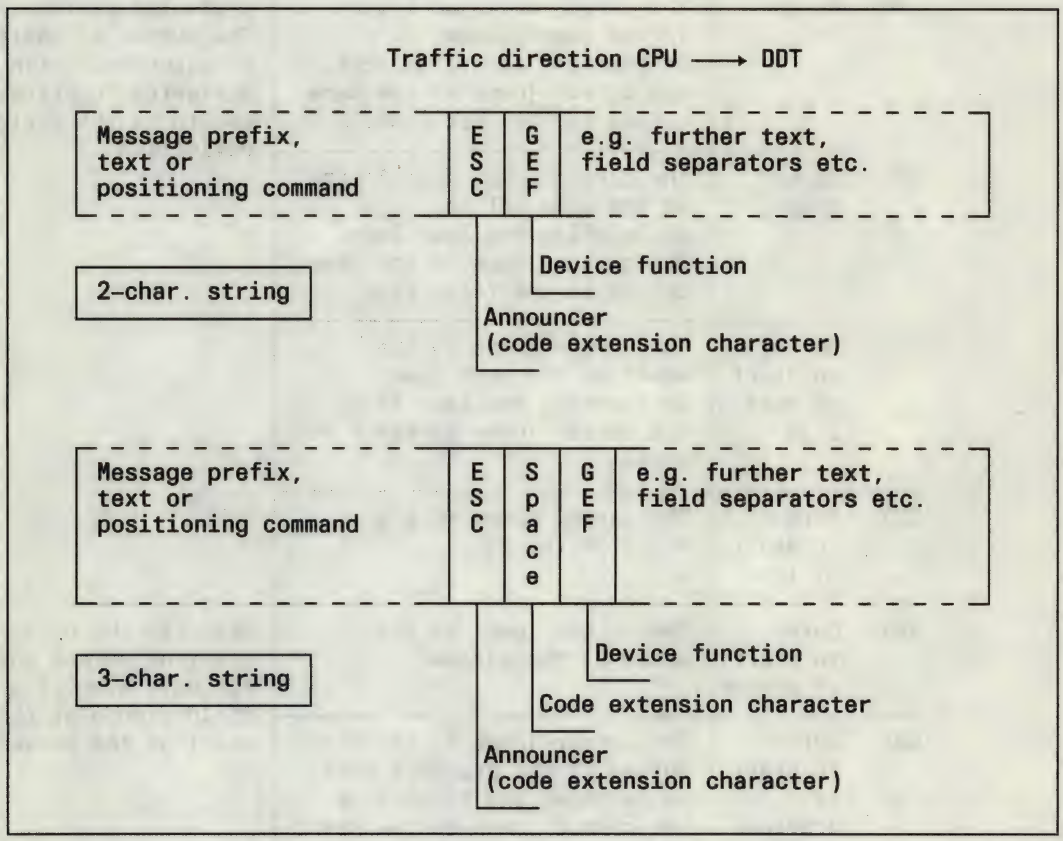

Traffic direction CPU ⟶ DDT
Message prefix, text or positioning command
ESC
GEF
e.g. further text, field separators etc.
Device function
2-char. string
Announcer (code extension character)
Message prefix, text or positioning command
ESC
Space
GEF
e.g. further text, field separators etc.
Device function
3-char. string
Code extension character
Announcer (code extension character)

Fig. 4-13 Character string with device functions

4.2.7 Description of Device Functions

All the following device functions can be initiated by means of a data output.

Cursor Positioning Functions

Basically, all positions on the screen can be accessed by means of the cursor positioning functions.

Function		Execution	
		Unformatted mode	Formatted mode
SMR	Cursor right	The cursor moves 1 position to the right in the write direction. On reaching the end of the screen, the cursor jumps to the start of the screen.	FST1 (cursor function SMF) and FST2 (ATAB) of PAR OOL SMF=0 and ATAB=1: Here the cursor skips those write positions in protected, non-markable fields. SMF=0 and ATAB=0, SMF=1 and ATAB=0, SMF=1 and ATAB=1: The cursor also accesses write positions in protected fields. The cursor is positioned in accordance with the activated function without regard to the field definition.
SML	Cursor left	The cursor moves 1 position to the left against the write direction. On reaching the start of the screen, the cursor jumps to the end of the screen.	
SMO	Cursor up	The cursor moves up 1 line in the same column. On reaching the first line, the cursor jumps to the same column in the last line.	
SMU	Cursor down	The cursor moves down 1 line in the same column. On reaching the last line, the cursor jumps to the same column in the first line.	
SNZ	Cursor to start of next line	The cursor jumps to the start of the next line. On reaching the last line, the cursor jumps to start of screen.	
SZA	Cursor to start of line	The cursor jumps to the start of the line.	
SBA	Cursor to start of screen	The cursor jumps to the start of the screen.	With SBA the cursor jumps to start of screen when FST4 bit 6=1, even if a protected field stands at the start of the screen.
SDZ	Cursor to start of previous line	The cursor jumps to the first column of the previous line. On reaching the first line, the cursor jumps to the start of the last line.	

Function		Execution	
		Unformatted mode	Formatted mode
TAR	Tabulator jump right	The cursor jumps 16 positions to the right in the write direction, always to columns 1, 17, 33, 49, 65, (81, 97,.. 113 and 129) of a line. On reaching the last column of the last line, the cursor jumps to the start of the screen.	The cursor jumps to the first column of the next modifiable field in the write direction.
TAL	Tabulator jump left	The cursor jumps 16 positions to the left against the write direction to columns (129, 113, 97, 81,) 65, 49, 33, 17 and 1 of a line. On reaching the start of the screen, the cursor jumps to the last column of the last line.	The cursor jumps to the first column of the next modifiable field against the write direction.

Table 4-16 Cursor positioning functions

Text Manipulation Functions

Function	Execution Unformatted mode	 Formatted mode
EFG Insert character	Starting from the cursor position, the characters are shifted in the write direction by the number of inserted characters. NULs present in the text are overwritten. If no further NULs are present, all following characters are shifted in the write direction. The function is effective beyond the end of the line. Characters shifted out beyond the end of the screen are lost. LZE handling: If during insertion an LZE character reaches the start of a physical line, the characters of this and the following lines will then be shifted in the write direction until the conditions conditions applying to the logical end of line are met. The EFG function is deactivated by means of 'LSP' and 'RS'. Inserting an LZE: The character at the cursor position and all characters to the right of it up to end of line are shifted to start of next line on inserting an LZE; the LZE is "inserted" at the cursor position. NUL characters are written to the remainder of the line. NUL characters between other characters in the following lines are overwritten. If there are insufficient NUL characters between the cursor position and the end of the display area, characters at end of screen are lost.	Operates as in unformatted mode, except that the EFG function remains restricted to an unprotected field. Characters shifted out beyond the end of the field are lost. Display control characters remain unaffected. When a character is inserted via the keyboard, the MOD bit is set in the FHC. Inserting an LZE: The character at the cursor position and all characters to the right of it up to end of field are overwritten with NUL if the field ends in the same line. If the field ends in a subsequent line, the characters starting at the cursor position are inserted at the start of the next line. The characters at the start of the next line are shifted to the right. NUL characters are overwritten here. If there are insufficient NUL characters between the start of the next line and the end of the field, characters at the end of the field will be lost. If the field contains further LZE characters, then only NUL characters may follow the LZE up to the end of the line or the end of the field. After insertion of an LZE, the cursor appears at the start of the next line if the field also occupies write positions in this line; otherwise it appears at the start of the next field.

Function		Execution	
		Unformatted mode	Formatted mode
AFG	Delete character	Deleting characters other than NUL: The character at the cursor position is deleted and the following characters up to the next NUL character or end of screen are shifted against the write direction. Positions vacated are occupied by NUL characters. Deleting a NUL: The next non-NUL characters to the right of the cursor up to the next NUL character or end of screen are shifted against the write direction. Positions vacated are occupied by NUL characters.	Operates as in unformatted mode, except that function ends at end of field of an unprotected field. When deleting via the keyboard, the MOD bit is set in the FHC.
EFZ	Insert line Lockable for the operator in char. FST1 of PAR OOL.	The characters in the line containing the cursor and the characters in the following lines are all shifted down by one line. When EFZ=0 the characters from the last line are lost. The line containing the cursor is erased (overwritten with NUL). When EFZ=1, the characters from the last line are copied to the line containing the cursor.	The characters in the line containing the cursor and the characters in the following lines are all shifted down by one line. The last line is copied to the line containing the cursor. When EFZ=0, the characters in the unprotected fields are then overwritten with NUL. When EFZ=1, the contents of the line remain unchanged. If all characters begin with field separators, the fields then retain their characteristics. If there are no field separators at the start of a line, then the characters up to the first start of field in this line assume the field characteristics of the last field of the previous line.

Function		Execution	
		Unformatted mode	Formatted mode
AFZ	Delete line Lockable for the operator in char. FST1 of PAR 001.	PAR 00L, FST1, bit 3=0: All characters starting from the line containing the cursor are shifted up. PAR 00L, FST1, bit 3=1: All characters starting from the line containing the cursor are shifted up. In addition, the last line is copied to the cursor line. (Scroll-up).	Operates as in unformatted mode. Formatted data is handled as follows: PAR 00L, FST1, bit 3=0: Field separators and protected data are copied from the cursor line to the last line. Unprotected data is replaced by NUL. PAR 00L, FST1, bit 3=1: Field separators and all data are copied from the cursor line to the last line.
RU	Roll-up function Lockable for the operator in char. FST1 of PAR 00L.	The characters from the second to the last lines are shifted up by one line. The characters from the first line are lost. NUL characters appear in the last line; the cursor is positioned at the start of the last line.	Operates as in unformatted mode. Field separators are also shifted. The last line contains no field separators. The field separators in the first line are lost.
RUB	Roll-up mode Can be switched on/off by specification in char. FST1 of PAR 00L.	After the last write position of the last line has been written to, the characters of the second to the last lines are all shifted up by one line. The characters from the first line are lost. NUL characters appear in the last line. The cursor is positioned at the start of the last line. When an LZE is entered in the last line - remainder of this line is overwritten with NUL - the Roll-up function is executed, - the cursor is positioned to the start of the new last line.	Operates as in unformatted mode, but including all field separators. *Note* When RU mode and ATAB are used, the last character present on the screen must be contained in an unprotected field.

Table 4-17 Text manipulation functions

Erase Functions

Function		Execution	
		Unformatted mode	Formatted mode
LSP	Erase memory contents Initiated by GEF1 of PAR 00D or by the LSP key (can be locked by specification in the FST2 of PAR 00L.	Following are erased: - from start of screen to end of screen; screen is overwritten with NUL. - PAR 00L - PAR 10L - PAR 70L (depending on FST2 of PAR 00L).	Operates as in unformatted mode, except that all field separators are also erased.
	Initiated by an ESC sequence in the text.	Erase from cursor to end of screen. The cursor position remains unchanged. Parameter areas are not erased.	Operates as in unformatted mode, except that all field separators are also erased. If this function begins at the start of a field, the field separators for this field are also erased.
LVD	Erase variable data	Erase from cursor to an EM inclusive or to end of screen (replaced by NUL characters). The cursor position remains unchanged.	When LVD=0, all characters, the MOD bit and the BLI bit are erased in unprotected fields from cursor to EM or to end of screen. In the field containing the cursor, the BLI bit is erased only if the field is also markable. When LVD=1, all characters, the MOD bit and the BLI bit are erased in unprotected fields from start of screen to EM or to end of screen. The cursor position remains unchanged.
		The erase limits to EM or end of screen are defined in PAR 00L, FST2: bit 6=0: from cursor to EM bit 6=1: start of screen to end of screen	

Function		Execution	
		Unformatted mode	Formatted mode
LZF	Erase to end of line or end of field	Erase from cursor to end of line or to an end marker (EM) inclusive. The cursor position remains unchanged.	All characters of an unprotected field are erased from cursor to end of field, or to an EM inclusive, in the same field. The cursor position remains unchanged. Handling of the MOD and BLI bits in the field separators is defined in PAR OOL, FST2, bit 5.
LVA	Erase variable data to address	All variable data from the current cursor position to the specified address is erased. The address specified in the string indicates the position of the first character not to be erased. If the specified address precedes the current write position, then all characters from the write position to end of screen and from start of screen to the specified address are overwritten with NUL. MOD bits remain unchanged. *Note* The function LVA can only be entered in the text segment. *Example:* To erase the variable data from start of screen to line 10, column 1: ESC-SBA-ESC-SP-LVA-ZLA-SPA1-SPA2 = X'2782274083A507F0' (coding in EBCDIC).	

Table 4-18 Erase functions

Data Transmission Functions

Function		Execution	
		Unformatted mode	Formatted mode
DÜ1	Initiate data transfer by means of function DÜ1	When a DÜ function is initiated, a send request (SAN) is made to the CPU; the SAN indicator lights up on the screen. SAN can be reset by means of RS. Transfer of the message takes place in accordance with the entries in character 1 (SAW1) of parameter area PAR 00L. The code of the send key used is transferred in character 7 (CDS) of PAR 00E.	
DÜ2	Initiate data transfer by means of function DÜ2	Function execution as under DÜ1. Transfer of the message takes place in accordance with the entries in character 6 (SAW2) of parameter area PAR 01L. The code of the send key used is transferred in character 7 (CDS) of the PAR 00E.	
K1 K2 K3	Initiate short message via the keyboard or from the CPU by means of the function Kx	Initiation with keys K1 – K3	
K4 K5 K6 K7 K8 K9 K10 K11 K12 K13 K14		Initiation with ESC key and one of the keys given here: V – K4 W – K5 M – K6 N – K7 O – K8 ? – K9 > – K10 = – K11 < – K12 ; – K13 : – K14 The code of the K function used is transferred in character 7 (CDS) of PAR 00E. The short message can comprise the PAR 00E, the PAR 01E, the PAR 02E and a text identifier. Initiation of a K function is possible also when the keyboard is locked and also in interactive mode after dialog has commenced. If a K function is to be initiated by the CPU, then the appropriate coding must be entered in character 1 (GEF1) of the PAR 00D.	

Function		Execution		Available with DDT
		Unformatted mode	Formatted mode	
F1 F2 F3	Initiate a data transfer by means of function Fx	Initiation with keys F1 - F3		All
F4 F5 F6 F7 . . . F20		The code of the F function used is entered in character 7 (CDS) of PAR 00E. The message on the screen and the message header are transmitted. If an F function is to be initiated by the CPU, then the appropriate coding must be entered in character 1 (GEF1 and possibly GEF2) of the PAR 00D.		All
F21 - F24		Initiation with keys CTRL + P1-4		9756, 9758, 9762, 9763

Table 4-19 Data transmission functions

Functions for Data Exchange between DDT and Centrally Connected Printer

Function		Execution		Available with DDT
		Unformatted mode	Formatted mode	
LA1 LA2 LA3 LA4 LA5 LA6 LA7	Local initiat. Initiat. LA1 - LA2 directly via the keyboard and LA3 - LA7 by ESC seq. or LA1 - LA7 by the CPU	Depending on the entries in the parameter areas PAR 10L/D-PAR 70L/D, a printer connected locally to the DDT and addressed in accordance with the parameter entries, or a printer terminal connected to the same cluster controller, is addressed.		All
		All characters from cursor to EM or end of screen are printed.	All fields defined as printable are printed from the cursor position to an end marker or to end of screen. Printer control characters in printable fields are interpreted in accordance with the specifications in the print command (PAR 10L/D-70L/D).	

Table 4-20 Functions for data exchange between DDT and centrally connected printer

Programmable Key Functions

Function		Execution Unformatted mode	Formatted mode
P	Turn P mode on/off	The P function activates/deactivates P mode. The P-areas can be loaded in P mode.	
P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 P11 P12 P13 P14 P15 P16 P17 P18 P19 P20	Select a P-area 1 - 20	STORAGE CAPACITY AND ENTRY OF CHARACTERS: Up to 16 characters can be entered in a P-area. 1) Text, device control and field separator characters (e.g. ESC; IS2, IS3, IS4 sequences) can be stored. The characters are not interpreted during loading. CHAINING OF P-FUNCTIONS: 1) In order to chain a number of P functions, the code of another P function can also be entered in a P-area, but the latter must have a greater weight than the P-area to be loaded. *Important:* The function 'DÜ' should be present only once. In character 4 (FST6) of PAR 01L, a USER LOCK can be entered: - for P1-P5 individually, - for P6-P20 jointly. In character 5 (FST7) of PAR 01L, a LOAD LOCK can be entered: - for P1-P5 individually, - for P6-P20 jointly. It is thus possible to reserve P-areas for the CPU. Loading of these P-areas from the keyboard is inhibited INITIATION BY THE CPU: Initiation by means of device control characters in the message or in the PAR 00D. If the P-area contains a function invalid during a data output, such as DÜ, K, F or LA, that function will not be interpreted and execution of the P function will be continued.	

1) With the 9755 DDT and up, the storage capacity is limited to 512 characters as the total of all the registers. The length of the individual registers is variable. Chaining is no longer meaningful.

Table 4-21 Programmable key functions

Special Functions

Function		Execution	
		Unformatted mode	Formatted mode
FAZ	Set field separators to initial state		The MOD bits (= bit 3) in all FHCs and the BLI bits (= bit 1) in the DCCs of the markable fields are erased. In character 3 (FST2) of PAR 00L it is also possible to specify whether this function is to be initiated only on receipt of the device function or automatically after each data output.
MAR	Mark a field	Input of 'MAR' is ignored. The Mark function is effective only in formatted mode.	Markable fields begin with FHC (bit 4=1) and DCC. A field can be marked when it contains the cursor and the MAR function is executed (MOD bit in the field separator and BLI bit in the DCC are set). The marked field starts to flash as a visual acknowledgment. Each further marking action deactivates or reactivates the function in PROTECTED FIELDS. With UNPROTECTED FIELDS, the MOD bit remains set.
RS	Reset	Resets the error indicators, send request (SAN) and operating states without changing the data memory contents.	
VA	Clear connection	ONLY FOR SINGLE TERMINALS (DDT-S): On completion of a current data transmission (DÜ) function, the connection is cleared down by the DDT.	
AM	Beginning marker	Unprotected fields (with NULs) inclusive of AM and character at cursor position are transmitted if the corresponding SAW is entered. The AM is interpreted only during data input. X'48' must be specified in character 1 (SAW1) of PAR 00L or in character 6 (SAW2) of PAR 01L. The last AM before the cursor is always interpreted. If the AM is missing, transmission takes place from start of screen. The AM has no effect at end of screen.	
EM	End marker	The end marker allows the following functions to be delimited: - printout of a message, - erase operation apart from LSP, - data input. FOR DATA INPUT, THE END MARKER MUST BE CONTAINED IN A TRANSMITTABLE FIELD.	
BEL	Visual alarm	When the control character BEL is received, 'BEL' flashes in the indicator line (audible alarm also sounds, if installed). BEL can be used, for example, as an output announcer. BEL is reset by means of the function BRS. *Note* This function is possible only with MSV.	

Function		Execution	
		Unformatted mode	Formatted mode
BRS	Reset BEL	When this device function is received, the 'BEL' indicator is reset.	
AKA	Audible alarm	When the control character AKA is received, an audible alarm (if installed) sounds and the 'BEL' indicator appears. EXAMPLE OF USE: The AKA is initiated at the end of a data output in order to draw the operator's attention to the end of output. THE 'BEL' INDICATOR IS RESET by means of function BRS or RS.	
SS	Set cursor	The function SS is used to store the current write position (line and column). This function cannot be initiated by the terminal user. If the message begins with the function SS, then after receipt of the message in the display area the cursor will again stand in its old position. If the message contains an SS function, then after receipt of the message the cursor will stand in the write position it occupied when this SS function was processed. If the message contains further SS functions, then after receipt of the message in the display area the cursor will stand in the write position it occupied when the last SS function was processed.	
WDH	Repeat	Duplicates a character up to a prespecified line/column address. The duplication acts in the same way as a data output of the corresponding number of characters. Field separators are overwritten when the function is initiated by the CPU. The function acts from the cursor position to the address specified under ZLA and SPA. If invalid addresses are specified, the function is not executed. If the specified address precedes the current write position, the repeat function acts beyond end of screen up to the specified address. If the specified address corresponds to the current write position, then the entire display memory will be filled with the predefined character. *Note* The function WDH can only be entered in the text segment.	
LZE	Logical end of line	The logical lines begin at the start of the line and end with LZE. They can be part of a line or also several lines long.	
		The cursor accesses all cursor positions. All characters after LZE are erased to the end of the line.	The cursor accesses all cursor positions. *Exception:* When ATAB=1 and cursor function SMF=0, the cursor skips protected, non-markable fields. All characters after LZE are erased to end of field or to end of line. Field separators are not erased.

Table 4-22 Special functions

4.3 Message Structure for Output of a System Message

The spatially separated lowermost line of the screen is referred to as the device indicator line in the initial state of the data display terminal (after powering on).

Character 5 (AZP) of PAR 00D allows the alternative display of a system line that can be used, for example, for the output of operator prompts:

- Character 5 (AZP) = X'40': Device indicator line is displayed.
- Character 5 (AZP) = X'41': System line is displayed.
- Character 5 (AZP) = X'00': Device indicator or system line is retained.

The system line is loaded by means of the output of a message from the CPU with special parameter entries.

160 characters can be stored:
(with 974x, 9750, 9751 and 9752 DDTs, up to 188 characters)

- up to 80 visible characters and
- up to 80 display control characters (with 974x, 9750, 9751 and 9752 DDTs, up to 48 DCCs).

Note

No DCC may follow the 80th character!

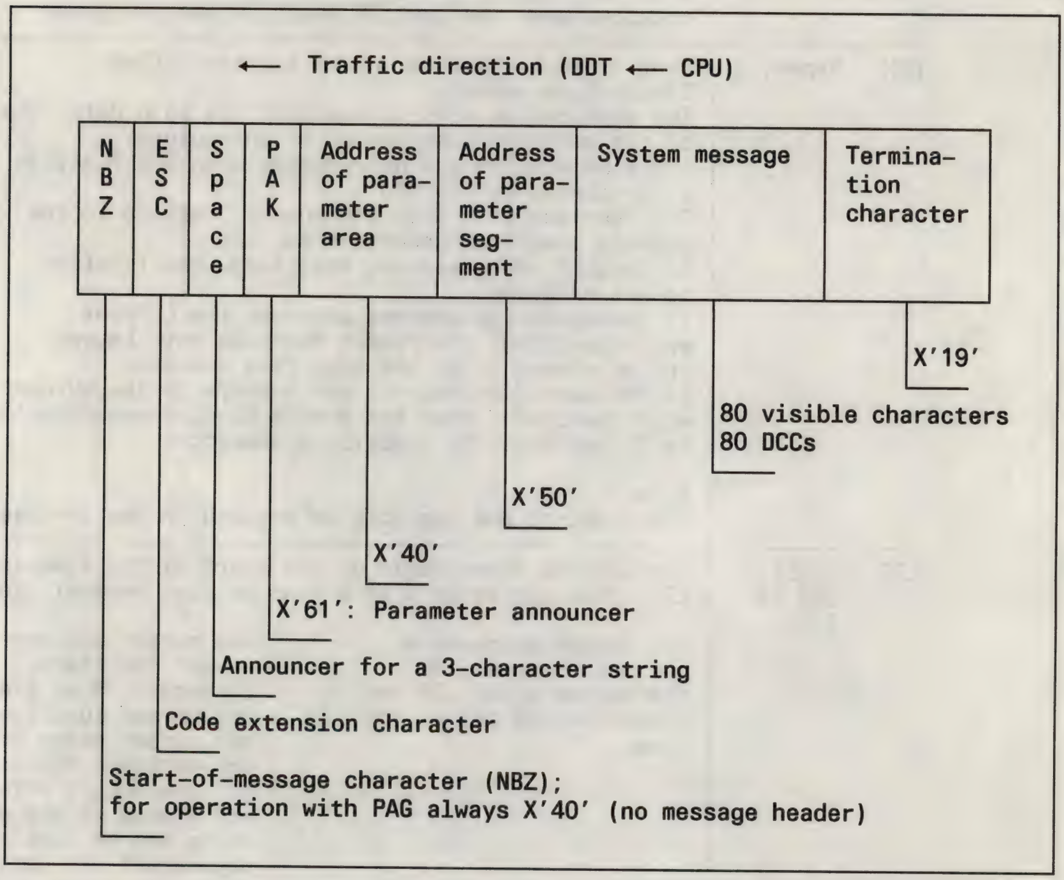

← Traffic direction (DDT ← CPU)
NBZ
ESC
Space
PAK
Address of parameter area
Address of parameter segment
System message
Termination character
X'19'
80 visible characters
80 DCCs
X'50'
X'40'
X'61': Parameter announcer
Announcer for a 3-character string
Code extension character
Start-of-message character (NBZ);
for operation with PAG always X'40' (no message header)

Fig. 4-14 Structure of message for system line

The end marker (`EM`) serves as a termination character for system messages and is not displayed in the system line. Following the end marker, any part of the system line remaining is erased with `NUL`.

`IS3` and `ESC` sequences must not occur in the system line.

Note

> It is not advisable to insert an audible alarm (`AKA`) or a visual alarm (`BEL`) when outputting the system line since this would cause the system line to be replaced by the device indicator line.

4.4 Operation with Status Messages

4.4.1 Query

Data can be requested on the configuration of the DDT, the monitor, the program version etc.

The status query may stand anywhere within the message, following the message header.

For system-dependent reasons, the query is output as part of an output message to the DDT. The query must not itself initiate a query reply. The query reply (status message) can be transmitted to the CPU, only in response to a query received, as part of an input message initiated by a DÜn, Fn or Kn function.

If a number of different queries are output in a single message, the corresponding query replies must then also be included with the next input.

4.4.2 Query Reply

The statuses of the components are given in the relevant query replies (status messages).

These apply to

- Configuration:
  The components present on this DDT.
- Monitor:
  The existing assignments, the selected screen format and the type of fonts.
- Program version:
  The revision level of the particular program.
- Display area:
  The current values for the specified display area.

Query replies must not be sent automatically as separate messages upon receipt of a status query. For system-dependent reasons, they may be transmitted only as part of an input message initiated by a DÜn, Fn or Kn function. If more than one query reply was requested in the last output message, then the requisite number of query replies must also be included with the next input.

The query replies may stand anywhere within the input message.

4.5 Message Structure for Operation with ID Card Reader

The ID card reader (AWL) allows controlled access to files protected by the operating system or by application programs.

Access should be permitted only on input of a user identification.

4.5.1 System Request for ID Card Information

The request bit in character 7 (FST8) of PAR 01L must be set and an input request must appear on the screen before reading and transmission of the ID card information can take place.

4.5.2 ID Card Format

ID cards in the ABA and SIPASS formats can be read by the ID card reader.

4.5.3 Reading ID Card Information

The format and display of the request are defined by the operating system or the application program.
When the request bit is set in PAR 01L, the keyboard is locked.
Any read attempt without a prior request issued by the CPU will result in the indicator GESPERRT (Inhibit) in the device indicator line.

When valid ID card information is read in, bit 1 = 1 is set in character 5 (ZZ1) of PAR 00E.
At the same time, the request bit in PAR 01L is reset by the data display terminal, the keyboard is released again and the ID card information is transmitted to the CPU.
Bit 1 of character 5 (ZZ1) of PAR 00E remains set until the ID card is removed from the reader.

After removal of the ID card, a K14 message is transmitted to the CPU and bit 1 of character 5 (ZZ1) of PAR 00E is reset to 0.

Note

> This bit must be interpreted by the application software in order to recognize removal of the ID card.

Reading Errored ID Card Information

If no ID card is present, then the read request from the CPU can be answered by means of keyboard input of a K14 message.
The K14 message is initiated by the ESC sequence ESC:.

The operating system or the application program can then reset the request bit in character 7 (FST8) of PAR 01L through the output of a message, and thus cause the keyboard to be released.

4.5.4 Error Messages

The ID card reader itself has no means of indicating errors.
Error messages are displayed as text in the device indicator line on the screen of the data display terminal.

Error text	Cause	Remedy
GESPERRT (Inhibit)	– An ID card was inserted into the reader although no request to do so (AWL indicator) existed.	Press RS key.
AUSWEISFEHLER (Card Error)	– The card was wiped too slowly through the reader. – The card was not wiped through the reader at an even speed.	Wipe the ID card through the reader again; if the error indicator still persists: enter 'ESC:'; the application will then determine whether or not continuation is possible.
	– The card is defective. – The card is invalid.	Use a new card.
	– The information on the card has a different data format.	The data format setting must be changed.
KARTE LEER (Card Empty)	– The card contains no information.	Use a new card.
	– The card was incorrectly inserted.	Wipe the card correctly through the reader.

Table 4-23 Error messages

4.6 Chip Card Terminal (CKT) Connection

The chip card terminal (CKT) can be connected to the data display terminal by means of an SS97 interface.

From the query reply that is transmitted to the CPU by the DDT during the next data input following output of a status query, it is possible to ascertain whether a driver for a CKT is loaded in the DDT.

The DDT does not constantly monitor whether the CKT is still connected.

Also, it does not report removal of the chip card or disconnection of the CKT.

If a CKT message is output to a DDT to which no CKT is connected, receipt of a CKT message will nevertheless be acknowledged by the DDT by means of an error message, if an error message was requested by the CPU.

If a CKT message cannot be transmitted to the CKT by the DDT despite repeated attempts, then an error message will occur if a monitoring interval is specified by the CPU for the command to the CKT.

If transmission of a CKT message from the DDT to the CPU is aborted after repeated attempts, no further action is however taken by the DDT. The CKT message is lost.

The DDT does not monitor whether the CKT transmits a message to the DDT.

The DDT expects in this case that the CKT will transmit a message to the DDT within the time permitted, even if no input is made by the user.

The DDT does not evaluate the contents of the data; it simply forwards the data to the CPU.

The keyboard is not locked on receipt of a CKT message.

The keyboard can only be locked/released by means of an appropriate entry in PAR 00L.

The AWL bit (ID card reader) must not be set in PAR 01L.

Initiation of K messages in the DDT is possible even when the keyboard is locked.

Note

> The DDT cannot prevent other messages (e.g. acknowledgment from a bypass print job), that do not come from the CKT, from being transmitted to the CPU following output of a CKT message to a CKT.

Transmission of messages between the CPU and a DDT is handled using a 7-bit character frame in the 'non-transparent mode' of the MSV1 procedure or the 'BAM' procedure. Some characters with codes between X'00' and X'1F' are used as data transmission control characters by these procedures.

The CKT is supported by the system as of BS2000 Version 18.

4.6.1 Data Transmission from CPU to CKT

The DDT recognizes a CKT message from entries in the parameter field of the CKT message.

In the case of errors in the parameter segment of the CKT message, processing of the message is aborted and - if requested - an error message is sent to the CPU.

Code conversion must be performed on the data in the data field of a CKT message before transmission from the CPU to the DDT (non-transparent transmission paths) if any component (e.g. 8211 RCG) on the transmission path permits only a 7-bit character frame for transmission. Whether or not code conversion has taken place must be specified in the parameter field.

The CKT driver in the DDT reverses the code conversion and then passes the data from the data field of the CKT message to the CKT without further evaluation of the data.

Coding in data field for transmission in expanded mode

Each two characters from the data field that are received are combined to form one character and passed to the CKT.

Each halfbyte of the 8-bit hex code must be converted into an ISO 7-bit code in accordance with the following rule before transmission from the CPU to the DDT.

2^7 2^2

High-order halfbyte	Low-order halfbyte

Hex code from central authorization terminal to the CKT

Hex halfbyte		ISO 7-bit code
X'0' ... X'9'	→	X'30' ... X'39'
X'A' ... X'F'	→	X'41' ... X'46'

The high-order halfbyte of the hex code is transmitted before the low-order halfbyte.

In the CKT driver in the DDT, each two characters of the ISO 7-bit code that are received are converted back into one 8-bit hex code character. The "initial hex code" reconstituted in this way is passed to the CKT.

4.6.2 Data Transmission from CKT to CPU

The data in the data field of the CKT message is converted by the CKT driver into a 7-bit code if any component on the transmission path permits only a 7-bit character frame for transmission.

Whether or not code conversion has taken place is specified in the parameter field.

Coding in data field for transmission in expanded mode

The data from the CKT is processed by the CKT driver and then inserted in the data field of the message.

Here each halfbyte of the 8-bit hex code from the CKT is converted into an ISO 7-bit code in accordance with the following rule for transmission:

2^7	2^2	
High-order halfbyte	Low-order halfbyte	Hex code from CKT device to central authorization terminal

HEX halfbyte		ISO 7-bit code
X'0' ... X'9'	→	X'30' ... X'39'
X'A' ... X'F'	→	X'41' ... X'46'

The high-order halfbyte of the hex code is transmitted before the low-order halfbyte.

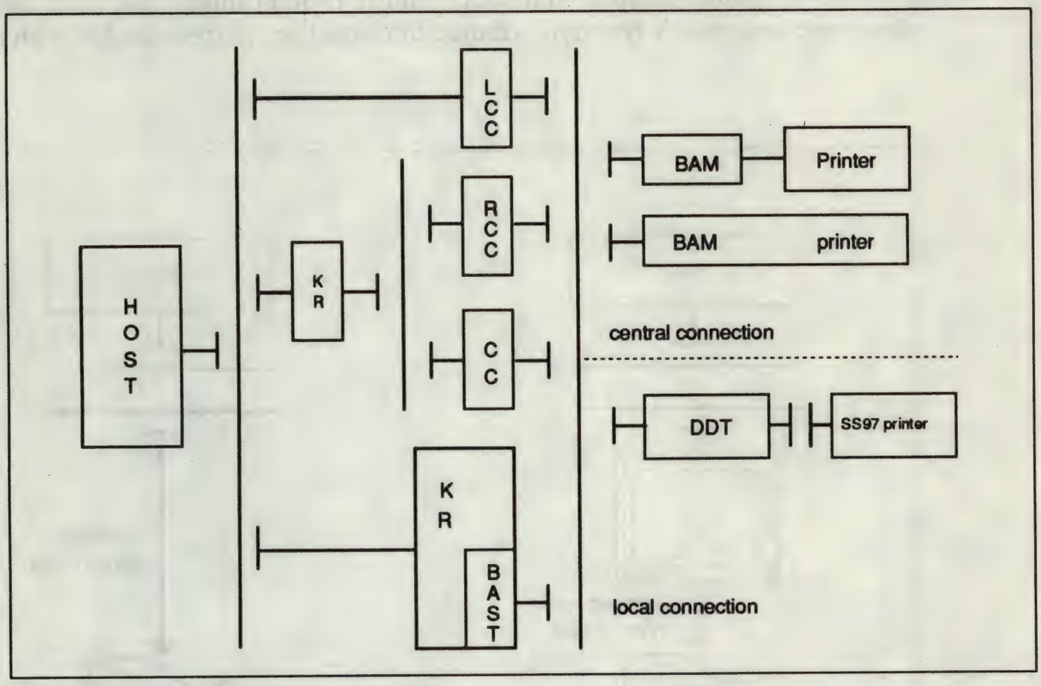
In the CPU, these ISO 7-bit characters must again be converted into an 8-bit hex code.

5 Printer Operation

5.1 Configuration

Printer connection is possible in the following basic configurations:

- centrally (printer with internal or external BAM controller)
- locally (printer with SS97 on a data display terminal)



BAM	bit-serial asynchronous multipoint line protocol
BAST	controller for BAM lines
CC	cluster controller
DDT	data display terminal
KR	communication computer
LCC	local cluster controller
RCC	remote cluster controller

Fig. 5-1 Printer connection options

5.2 Operating Modes

A distinction is made between the following two operating modes for printers:

- unsolicited print output and
- hardcopy mode.

Unsolicited Print Output

The data to be printed comes from an application and is intended exclusively for a printer. All printer control characters required for printing (line feed, carriage return, sheet ejection, switch typestyle, change line spacing ...) are included with the data.

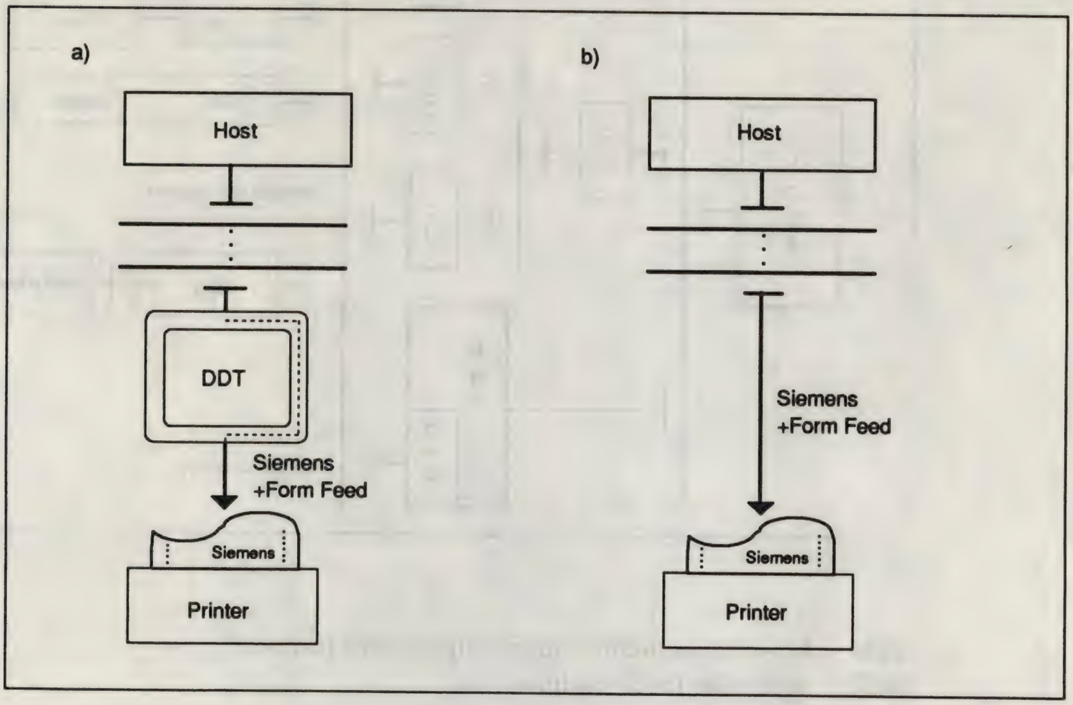


Fig. 5-2 Unsolicited print output with
a) local connection (bypass mode)
b) central connection

Hardcopy Mode

In hardcopy mode the screen contents are transferred with unchanged line format to the printer.
The screen contents are printed from the cursor up to an end marker. If no end marker is present, printing continues to end of screen.
Control characters for new lines are generated by the data display terminal.

With the 9755, 9756, 9758, 9762 and 9763 DDTs, a control character for form feed or sheet change can also be passed to the printer in hardcopy mode.

Control characters for the printer that are contained in the text are replaced with blanks by the data display terminal.

Hardcopy mode can be further subdivided into:

- manual hardcopy mode and
- programmed hardcopy mode.

In **manual hardcopy mode** the text is first displayed on the DDT. The printout is initiated by means of an LAx key. The text may either come from the CPU or have been entered by the user at the keyboard.

In **programmed hardcopy mode** the message is also displayed on the DDT, but the printout is initiated automatically. Keyboard input is not possible.

Printing does not take place until message transfer to the data display terminal has been completed.

The print output is always controlled by the data display terminal.
The message header, data display terminal parameter areas and output message are constructed only for the data display terminal.

5.3 Acknowledgments

DDT and BAM controller

With unsolicited print outputs, **positive acknowledgments** provide information to the effect that

- a particular message has been printed in its entirety (line printer)
- a particular message has been sent in its entirety to the printer (page printer).

If either of these actions was not performed successfully, a **negative acknowledgment** provides relevant information as to the cause.

BAM controller only

A **power failure acknowledgment** provides information as to whether the BAM controller has been switched on and whether a voltage drop has occurred in the controller.
If a power failure acknowledgment occurs during operation, all messages that have not been positively acknowledged are lost and must be repeated. Parameter settings for the printer are deleted and must also be repeated.

Table of Functions for Data Display Terminals

Function	Section	97. Data Display Terminal							
		4x	50	52	55	56	58	62	63
Operation with message header 1)	4.1	x	x	x	x	x	x	x	x
Message section with PAG	4.2.1	x	x	x	x	x	x	x	x
Message section with structured commands	4.2.2								
LSID	4.2.2.1					x	x	x	x
LPBE	4.2.2.2					x	x	x	x
LZEG	4.2.2.3								x
DEBS	4.2.2.4								x
RDSS	4.2.2.5								x
SRBR	4.2.2.6								x
Query / Query reply	4.4.1/2						x	x	x
Mess.sect. with position. command	4.2.3	x	x	x	x	x	x	x	x
Message section with fields seps.	4.2.4	x	x	x	x	x	x	x	x
Mess.sect. with field & char.atts	4.2.5								x
Mess.sect. with device functions	4.2.6/7								
Cursor positioning operations	4.2.7	x	x	x	x	x	x	x	x
Text manipulation functions	4.2.7	x	x	x	x	x	x	x	x
Erase functions	4.2.7	x	x	x	x	x	x	x	x
Data transmission functions	4.2.7								
DÜ1, DÜ2, K1 - K14, F1 - F20	4.2.7	x	x	x	x	x	x	x	x
F21 - F24	4.2.7					x	x	x	x
Data exchange DDT and centrally connected printer (LAx)	4.2.7	x	x	x	x	x	x	x	x
Programmable key functions	4.2.7	x	x	x	x	x	x	x	x
Special functions	4.2.7	x	x	x	x	x	x	x	x
Output of a system message	4.3								
with up to 48 DCCs	4.3	x	x	x					
with up to 80 DCCs	4.3				x	x	x	x	x
Operation with ID card reader	4.5	x	x	x	x	x	x	x	x
CKT connection	4.6								x

1) The use of message headers during data output is not permitted for new applications.

Abbreviations

AD1/2	address 1/2
ADA	print command
AFP	alternating pulse edge transmission method
AM	beginning marker
ATAB	automatic tabulator
AUSG	program revision level
AWL	ID card reader
AZP	indicator line
BAM	bit-serial asynchronous multipoint line protocol
BAST	controller for BAM lines
BLI	flash, blink
CC	cluster controller
CDS	code for send keys
CFEP	compact front-end processor
CKT	chip card terminal
CPU	computer
CR	carriage return
DCC	display control character
DCE	data circuit-terminating equipment
DCM	Data Communication Methods
DDT	data display terminal
DDT-C	data display terminal for cluster connection
DDT-S	data display terminal for standalone connection
DEAA	define display attribute
DEBS	define screen
DEFA	define color attributes
DEZS	define font
DRU	printable
DÜ	data transmission, data communication
EA	selecting
EBCDIC	Extended Binary-Coded Decimal Interchange Code
EM	end marker
ESC	Escape
EZFA	create field attributes
FAT	field attribute
FAZ	reset field separator to initial state
FB	field mode
FEM	error message
FEP	front-end processor
FEPR	remote front-end processor
FHC	field handling character
FHS	Format Handling System
FST	function control character
FTZ	field separator
GEF	device function
GEF1/2	device function 1/2
GNU	protected non-transmittable
GÜ	protected transmittable
ID1/2	identification character 1/2

IE	interface expander
IFEP	integrated front-end processor
ISO	International Organization for Standardization
ISx	information separator
ITC	integrated terminal controller
KAN	channel address
KR	communication computer
LAP1	LA functions parameter
LAx	local initiation (printer)
LCC	local cluster controller
LOGZEG	logical character generator (symbolic name for ZEG)
LPBE	read P-areas
LSID	read SIDATA parameters
LZE	logical end of line
LZEG	load character generator
MAR	markable
MCR	magnetic card reader, ID card reader
MOD	modification
MOFA	modify field attributes
MSC	medium-speed concentrator
MSV1	procedure: Medium Speed Variant 1
MULTIKOS	multi-component terminal support
NBZ	start-of-message character
NT	message text
NUM	numeric
PA	parameter area
PAG	parameter entries
PAK	parameter announcer
PAR	parameter
PBH	parameter handling
PCC	protocol converter concentrator
PDN	program system for teleprocessing and network control
PHYZEG	physical character generator
PLUS	load and store program keys
PS	parameter segment
RB1/2	return bytes 1/2
RCC	remote cluster controller
RDSS	reset screen
RM	acknowledgment
RS	reset
RSZA	reset character attribute
RU	roll-up
RUB	roll-up mode
SA	polling
SAW	send command
SMF	cursor function
SP	Space
SPA	column address
SRBR	write display area
SS	interface
STA	structured command
STB	status byte
STZ	physical control character
TC	terminal computer

TI	text identifier
TID	terminal identification
TIL	text identifier length
TKL	text header length
UNT	underscored
WAR	queue identifier
WTÜ	coded di-phase transmission method
ZAT	character attribute
ZEG	character generator
ZLA	line address
ZZx	status character

References

[1] TRANSDATA
Data Display Terminals
Code Tables

Target group
Programmers

Contents
Codes required for programming of DDTs and for data interchange between printers

Applications
Application software programming

Ordering manuals

The manuals listed above and the corresponding order numbers are to be found in the *List of Publications: Data Systems*, which also tells you how to order manuals. New publications are listed in the *Druckschriften-Neuerscheinungen Datentechnik (New Publications).*

You can arrange to have both of these sent to you regularly by having your name placed on the appropriate mailing list. Your local Siemens office will help you.

Index

$DIALOG 1-3

ABA 4-61
acknowledgment
– requests 4-5
AFP 3-1
alternating pulse edge method (AFP) 2-1
announcer 4-11, 4-30, 4-31, 4-32, 4-44, 4-45
attributes 4-31

BAM 2-1, 3-1
base parameter 4-7
basic
– functions 1-3
– information 4-6
buffer length 4-3
bypass mode 4-8

card error 4-62
channel address 4-8
character
– attribute 4-33, 4-39, 4-40, 4-41
– box 4-16, 4-17, 4-18, 4-19, 4-21, 4-23, 4-26
chip card terminal (CKT) 4-63
cluster
– connection 3-1
– controller 3-2, 4-8
– terminals 2-1
code extension character 4-7
color attribute 4-42
column address 4-30
compatibility 1-3
concentrator 1-1, 2-1
connection options for data display terminals 2-1
CPU 4-11
cursor 4-30
– function (SMF) 4-9
– position 4-9
– positioning function 4-44, 4-46

data
– display terminal 2-1, 4-2, 4-4, 4-6, 4-9, 4-11
– display terminals 1-3
– exchange 4-11
– input 3-3, 4-2, 4-4, 4-8, 4-9, 4-10
– output 3-3, 4-1, 4-4, 4-9, 4-11, 4-31
data transmission 4-8
– function 4-44
– functions 4-44, 4-53
– key 4-45
Datex network 2-1
DDT-C 3-2
define screen 4-25
device function 4-6, 4-8, 4-44, 4-45, 4-46
– character 4-9
– character 1 4-11
– character 2 4-11
device
– indicator line 4-58
– type 1-3
display
– attribute 4-33, 4-40, 4-42
– characteristics 4-35
– control character 4-31, 4-32

end marker 4-59
erase function 4-44, 4-51
error message 4-4, 4-29, 4-62

field and character attributes 4-6, 4-33
field attribute 4-33f, 4-38, 4-39
– 1 4-34
– 2 4-36
– 3 4-36
– 4 4-36
field
– characteristics 1-4, 4-37
– handling character 4-10, 4-31, 4-32, 4-36
– mode 4-2, 4-8, 4-9
– protected 1-4
– separators 4-6, 4-9, 4-30, 4-31, 4-32, 4-38, 4-45
– unprotected 1-4
font 4-36, 4-38, 4-43
formatted mode 1-4, 4-10
function control 4-8
– character 1 4-9
– character 2 4-9
– character 3 4-9

hardcopy mode 5-3
HDLC 3-1

ID card 4-61
– format 4-61
– information 4-10
– input 4-61
– reader 4-61
indicator line 4-11
inhouse line 3-2
interactive mode 4-9
interface expander 2-1

key function 4-44, 4-55

line
– address 4-30
– length 4-3
load
– character generator 4-14
– lock 4-10

local
- connection 3-2
- line controller 3-1
long-distance line 3-2
LSP 4-6

message
- header 4-1, 4-2, 4-6, 4-9, 4-44
- header length 4-8
- prefix 4-1, 4-2, 4-5, 4-32, 4-44
- section 4-6
- structure 4-1
- text 4-1, 4-7
- texts 4-6
- traffic 3-1
MSV1 3-1

operating
- modes 1-3
- states 4-6
- system PDN 1-2

P-areas 4-13
PAG sequence 4-9
PAR
- 00E 4-2
- 00L 4-8
- 01E 4-2, 4-4
- 02E 4-2
- 03E 4-2
parameter
- announcer 4-7
- area 4-7, 4-10, 4-11
- areas 4-2, 4-6, 4-7, 4-8
- entries 4-1, 4-6, 4-7, 4-9, 4-10, 4-44
- entries structure 4-7
- segment 4-7
- segments 4-6, 4-8
polling 3-1
positioning command 4-6, 4-30
possible errors 4-9
print command 4-8
printer
- operation 5-1
- terminal 4-8

public leased-circuit data network 2-1

query 4-60
- reply 4-60
queue identifier 4-3, 4-9

remote connection 3-2
reset screen 4-27

selecting 3-1
send command 4-3, 4-6, 4-9, 4-10
SIDATA parameters 4-13
SIPASS 4-61
software components 1-2
special function 4-44, 4-56
standalone
- connection 3-1
- terminal 2-1
start-of-message character 4-1, 4-7
status character
- 1 4-3
- 2 4-3
structured commands 4-6, 4-12
switched telephone network 2-1
system
- line 4-8, 4-58
- message 4-58

tabulator 4-9
terminal
- computer 1-1
- identification 4-2
termination character 4-7, 4-9
text
- handling 4-8
- identifier 4-2, 4-4, 4-8
- identifiers 4-6
- manipulation function 4-44, 4-48
TRANSDATA 1-1
- hardware 1-1
- software 1-2

unformatted mode 1-3
unsolicited print output 5-2
usage lock 4-10

write display area 4-27

From

Name ____________________

Company ____________________

Street ____________________

City/Postal Code ____________________

Phone ____________________

Date ____________________

To
Siemens AG
DI ST QM 2
Manualredaktion
Otto-Hahn-Ring 6

8000 München 83

Reader's Reply Form

TRANSDATA Data Display Terminals Functional Description
User's Guide
Edition July 1990

Page	Criticisms / Suggestions / Corrections

Page	Criticisms / Suggestions / Corrections

From

Name ______________________________

Company ______________________________

Street ______________________________

City/Postal Code ______________________________

Phone ______________________________

Date ______________________________

To
Siemens AG
DI ST QM 2
Manualredaktion
Otto-Hahn-Ring 6

8000 München 83

Reader's Reply Form

TRANSDATA Data Display Terminals Functional Description
User's Guide
Edition July 1990

Page	Criticisms / Suggestions / Corrections

Page	Criticisms / Suggestions / Corrections